Sustainability is the main challenge faced by humanity today on global and local scales. This dissertation investigates how the latest developments within computer science and ICT can be applied to establish participatory, low-cost tools and practices that enable citizens to monitor, raise awareness about, and contribute to the sustainable management of the commons they rely on, and thereby protect or improve their quality of life.

The general approach proposed in this work is to use *community memories* – as central data repositories and points of interaction for community members and other stakeholders – and the novel combination of *participatory mobile sensing* and *social tagging* – as a low-cost means to collect quantitative and qualitative data about the state of the commons and the health, well-being, behaviour and opinion of those that depend on it.

This work also demonstrates in detail how this approach can be turned into a concrete solution for the problem of *environmental noise* – commonly referred to as noise pollution. The *NoiseTube* system presented here enables a participatory, low-cost approach to the assessment and mapping of environmental noise and its impact on citizens' quality of life. The system has been operational since May 2009 and has since been iteratively improved and validated through lab and field experiments. Via coordinated campaigns with volunteering citizens it is established that participatory noise mapping is a suitable alternative for, or a valuable complement to, conventional methods applied by authorities. NoiseTube is the completest and most widely used participatory noise mapping solution to date.

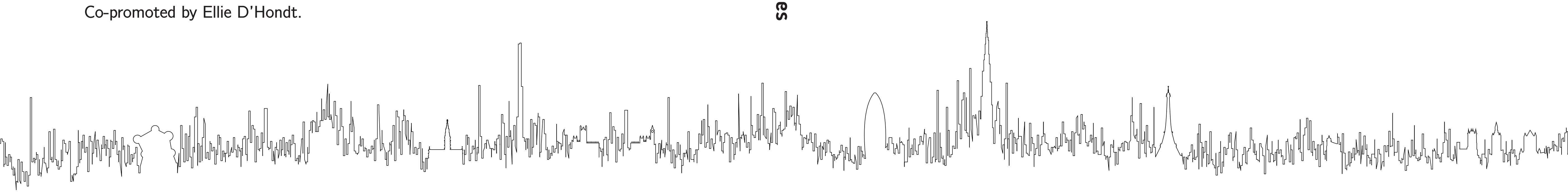
PhD thesis by Matthias Stevens, Vrije Universiteit Brussel, June 2012.
Promoted by Luc Steels.
Co-promoted by Ellie D'Hondt.

Matthias Stevens – Community memories for sustainable societies

Community memories for sustainable societies

The case of environmental noise

Matthias Stevens



Vrije Universiteit Brussel

WWW.ASPEDITIONS.BE
9 789057 181283

VUBPRESS

Vrije Universiteit Brussel

Faculty of Science and Bio-Engineering Sciences
Department of Computer Science

Community memories for sustainable societies
The case of environmental noise

Graduation thesis submitted with intention to obtain the degree of PhD in Science

Matthias Stevens

Promotor: Prof. Dr. Luc Steels
Copromotor: Dr. Ellie D'Hondt

Brussels, June 2012

Vrije Universiteit Brussel

Faculteit Wetenschappen en Bio-ingenieurswetenschappen
Vakgroep Computerwetenschappen

Community memories for sustainable societies
The case of environmental noise

Proefschrift ingediend met het oog op het behalen van de graad van Doctor in de Wetenschappen

Matthias Stevens

Promotor: Prof. Dr. Luc Steels
Copromotor: Dr. Ellie D'Hondt

Brussel, Juni 2012

2012 Uitgeverij VUBPRESS Brussels University Press
VUBPRESS is an imprint of ASP nv (Academic and Scientific Publishers nv)
Ravensteingalerij 28
B-1000 Brussels
Belgium
Tel.: +32 (0)2 289 26 50
Fax: +32 (0)2 289 26 59
E-mail: info@vubpress.be
www.vubpress.be

ISBN 978 90 5718 128 3
NUR 980 / 962 / 907
Legal deposit D/2012/11.161/064

Cover design by Jasper Stevens.
Printed by Silhouet, Maldegem, Belgium.

The author can be contacted at: mstevens@vub.ac.be.
A digital version of this book, as well as any addenda/errata, can be found at:
www.brussense.be/phd-matthias.

To An, for everything.

Abstract

Sustainability is the main challenge faced by humanity today on global and local scales. Most environmental problems can be seen as the tragic overexploitation of a *commons*. In this dissertation we investigate how the latest developments within computer science and ICT can be applied to establish participatory, low-cost tools and practices that enable citizens to monitor, raise awareness about, and contribute to the sustainable management of the commons they rely on, and thereby protect or improve their quality of life.

As a general approach we propose the use of *community memories* – as central data repositories and points of interaction for community members and other stakeholders – and the novel combination of *participatory mobile sensing* and *social tagging* – as a low-cost means to collect quantitative and qualitative data about the state of the commons and the health, well-being, behaviour and opinion of those that depend on it.

Through applied, interdisciplinary research we develop a concrete solution for a specific, socially relevant problem, namely that of *environmental noise* – commonly referred to as noise pollution. Under the name *NoiseTube* we present an operational system that enables a participatory, low-cost approach to the assessment of environmental noise and its impact on citizens' quality of life. This approach can be applied in the scope of citizen- or authority-led initiatives. The NoiseTube system consists of a sensing application – which turns mobile phones into a sound level meters and allows users to comment on their experience via social tagging – and a community memory – which aggregates and processes data collected by participants anywhere. The system supports and has been tested and deployed at different levels of scale – *personal*, *group* and *mass sensing*. Since May 2009 NoiseTube has been used by hundreds, if not thousands, of people all around the world, allowing us to draw lessons regarding the feasibility of different deployment, collaboration and coordination scenarios for participatory sensing in general. While similar systems have been proposed ours is the completest and most widely used *participatory noise mapping* solution to date. Our validation experiments demonstrate that the accuracy of mobile phones as sound level meters can be brought to an acceptable level through calibration and statistical reasoning. Through coordinated NoiseTube campaigns with volunteering citizens we establish that participatory noise mapping is a suitable alternative for, or a valuable complement to, conventional methods applied by authorities.

Samenvatting

Duurzame ontwikkeling is de belangrijkste uitdaging waarmee de mensheid momenteel geconfronteerd wordt op globaal en lokaal vlak. De meeste milieuproblemen kunnen gezien worden als de tragische overexploitatie van een *commons*. In deze dissertatie onderzoeken we hoe de nieuwste ontwikkelingen binnen de computerwetenschappen en ICT toegepast kunnen worden om participatieve, goedkope systemen en methoden te ontwikkelen welke burgers in staat stellen om toezicht te houden op, aandacht te eisen voor, en bij te dragen aan het duurzaam beheer van de commons waar ze afhankelijk van zijn, en zodoende hun levenskwaliteit te beschermen of te verbeteren.

Als een algemene aanpak stellen het gebruik voor van *community memories* – als centrale vergaarplaatsen voor gegevens en interactiepunten voor burgers en andere belanghebbenden – en de innovatieve combinatie van *participatory mobile sensing* en *social tagging* – als een goedkope manier om kwantitatieve en kwalitatieve gegevens te verzamelen over de toestand van de commons en de gezondheid, het welzijn, het gedrag en de opinie van zij die ervan afhangen.

Doormiddel van toegepast, interdisciplinair onderzoek ontwikkelen we een concrete oplossing voor een specifiek, maatschappelijk relevant probleem, namelijk dat van *omgevingslawaai* – beter bekend als geluidshinder. We presenteren *NoiseTube*, een operationeel systeem dat een participatieve, goedkope aanpak voor het in kaart brengen van omgevingslawaai en de impact op de levenskwaliteit van burgers mogelijk maakt. Deze aanpak kan toegepast worden in het kader van initiatieven geleid door burgers of overheden. Het NoiseTube systeem bestaat uit een meetapplicatie – welke mobiele telefoons verandert in geluidsmeters en gebruikers toelaat om commentaar te geven over hun ervaring via social tagging – en een community memory – welke instaat voor het aggregeren en verwerken van gegevens verzameld door deelnemers overal. Het systeem ondersteunt en is getest en ingezet op verschillend schaalniveaus – *personal*, *group* en *mass sensing*. Sinds mei 2009 is NoiseTube gebruikt door honderden, zo niet duizenden, mensen over heel de wereld, wat ons in staat stelt om lessen te trekken met betrekking tot de haalbaarheid van verschillende uitrol-, collaboratie- en coördinatiescenario's voor participatory sensing in het algemeen. Hoewel er vergelijkbare systemen voorgesteld zijn is onze oplossing voor *participatieve geluidskartering* de compleetste en wijdst gebruikte tot op heden. Onze

validatie-experimenten tonen aan dat de nauwkeurigheid van mobiele telefoons als geluidsmeters tot op een aanvaardbaar niveau gebracht kan worden door middel van kalibratie en statistische redenering. Doormiddel van gecoördineerde geluidsmeetcampagnes met vrijwilligers staven we dat participatieve geluidskartering een geschikt alternatief of een waardevolle aanvulling is voor conventionele methoden gebruikt door overheden.

Acknowledgements

Since I started out as a PhD student in October 2007, I have been fortunate to be surrounded, both at work and in life, by a number of fantastic people, some of whom I knew from before, others whom I met in the course of my work or the places it took me. Without the direct or indirect support of these people I would never have completed this dissertation. Therefore I would like to express my gratitude to all of them.

First and foremost I would like to thank my advisors Prof. Dr. Luc Steels and Dr. Ellie D'Hondt for respectively promoting and co-promoting this doctoral thesis. I also want to thank the other members of my jury, Prof. Dr. Viviane Jonckers, Prof. Dr. Beat Signer, Prof. Dr. Caroline Pauwels, Prof. Dr. Muki Haklay and Prof. Dr. Catherine Lavandier, for agreeing to read, comment on and judge my thesis.

I would also like to thank Luc Steels for inviting me, back in 2008, to do an internship at the Sony Computer Science Lab in Paris, which has been the starting point for the work discussed in this dissertation. At Sony CSL Paris I have had the opportunity to work with highly talented and pleasant people. I am especially grateful to Nicolas Maisonneuve, who played a leading role in the NoiseTube project until late 2009 and with whom I have closely collaborated during that time. Nicolas' creativity, enthusiasm and hands-on approach to software development have had a lasting influence, not only on tools that we built but also on my vision about research and innovation. Moreover I would like to thank Dr. Peter Hanappe for his advice and quite simply for being one of the nicest people I have ever come across. Finally I want to thank Dr. Maria Niessen and Bartek Ochab, who were involved in the NoiseTube project for shorter periods.

Since late 2009 I have been fortunate to work closely with Ellie D'Hondt. Together we formed BrusSense, a new research team hosted by the Software Languages (SOFT) and Artificial Intelligent (AI) labs of the VUB's Computer Science department. I want to thank Ellie for our productive collaboration and for guiding me towards and through the thesis writing process as well as for commenting on the text. In the context of BrusSense I also enjoyed working with master student Sander Bartholomees (now graduated). Moreover I want to thank Marleen Wynants and Sara Engelen of VUB CROSSTALKS for providing us with a dissemination platform on numerous occasions.

Acknowledgements

I would like to thank Prof. Dr. Theo D'Hondt and Prof. Dr. Wolfgang De Meuter for going through the trouble of convincing me to do a PhD (and to stick with it), and to regularly challenge me with critical questions. Additionally I want to thank Theo, who was my initial promotor, for the almost absolute freedom he gave me to find my own research topic, even though that let me away from the core themes of the PROG/SOFT lab and to some extent away from the lab itself. While hugely challenging and confronting at first, this freedom, in combination with the direction and opportunity provided by Luc Steels, is what has allowed me to develop myself as a researcher and to effectively open up a new research track within PROG/SOFT and the Computer Science department in general. Moreover I want to thank Wolfgang, to whom I was a teaching assistant for 4 years, for his candour, enthusiasm and sense of perfectionism – similar to, yet regularly in disagreement with, my own – which helped to make my teaching duties an enjoyable, challenging and gratifying experience, rather than just a distraction from my research.

Coming to work would not have been as enjoyable, nor as productive, if it was not for the companionship and friendship of my colleagues – during and after working hours. I want to specifically, but not exclusively, thank: Andoni, Andy, Carlos, Christophe, Coen, Dries, Elisa, Engineer, Ellie, Isabel, Jorge, Kevin, Lode, Nicolás, Stefan, Stijn T., Thomas, Tom, Yves and Wolfgang, of the VUB SOFT and former PROG lab; Beat of the VUB WISE lab; Joachim, Joris, Katrien and Pieter, of the VUB AI lab; and Amaury, François, Gabriele, Gonçal, Katya, Maria, Michael, Nicolas, Pierre, Peter, Remi, Simon, Sophie and Vanessa, of Sony CSL Paris.

Most of all I want to thank my girlfriend An, without whose constant encouragement, loyal support, patience and love I would probably not have found the courage to complete this gargantuan task. I also want to thank her for acting as a sounding board, providing insightful comments, proofreading, keeping me well-fed, and tolerating my ever-longer hair.

I am grateful to my parents, my brother Jasper and his girlfriend Anouck, for their support and for bearing with my grumpy replies to their well-meant enquiries and comments on my progress. I also want to thank my dad for proofreading a few chapters and Jasper for designing the beautiful cover for this dissertation.

I want to thank my friends in Ghent, Brussels, Paris or thereabout, for their support and for bearing with me during my ±1 year hermit existence. I especially want to mention: Andreas, Anouck, Axel, Barbara, Bert, Carole, Caroline, Charles, David, Doruntina, Eli, Els, Erwin, Gwen, Gwenola, Hannah, Jasper, Jonas, Julie, Katrien B., Katrien D., Koen, Laure, Lauwke, Lieselotte, Margot, Marianne, Marie-Laure, Matthias, Mieke, Nathalie D., Nathalie R., Olmo, Pieter B., Pieter V. P., Petra, Rien, Samuel, Simon B., Simon D., Sofie, Stéphane, Thomas, Tom, Trien, Valérie, Wendy, Wim, Wouter and Yole.

Last but not least, in keeping with the tradition started in my licentiate's thesis, I would like to thank my cat Puck for always cheering me up.

Contents

List of Figures

List of Charts

List of Tables

List of Boxes

List of Acronyms & Abbreviations

2G	2nd generation mobile telecommunications
3D	3-dimensional
3G	3rd generation mobile telecommunications
4G	4th generation mobile telecommunications
AAC	MPEG-2/4 Advanced Audio Coding, an audio encoding format
AAF	Android Application Framework (see section E.4.2)
ACL	Android Core Libraries (see section E.4.1)
ADC	Analogue-to-digital converter, also A/D (see section A.4.2)
AIFF	Audio Interchange File Format, a container format
ANSI	American National Standards Institute
API	Application programming interface
app	Application
ASA	Acoustical Society of America
ASP	Analogue signal processor
CD	Compact Disk
CDC	Connected Device Configuration [508] (see section E.3.1)
CDDA	Compact Disk Digital Audio
CEO	Chief executive officer
CLDC	Connected Limited Device Configuration [509] (see section E.3.1)
CM	Community memory (see section 2.5)
CPR	Common-pool resource (see section 2.3.1.2)
CPU	Central processing unit
CRUD	Create, Read, Update, Delete – database record operations
CSCW	Computer-supported cooperative work
CSS	Cascading Style Sheets
CTO	Chief technology/technical officer
DAB	Digital Audio Broadcasting
DAC	Digital-to-analogue converter, also D/A (see section A.4.2)
DALY	Disability-adjusted life year

DBMS	Database management system
DGPS	Differential GPS [580]
DIY	Do-it-yourself
DSP	Digital signal processor
DVB	Digital Video Broadcasting
DVD	Digital Versatile/Video Disc
EC	European Commission
ECG	Electrocardiogram
EMEA	Europe, the Middle East and Africa
END	The EU's Environmental Noise Directive [173]
EQLS	European Quality of Life Survey [20]
ESP	Effective sound pressure (see section A.2.2)
EU	European Union
EU-15	The 15 countries that made up the EU prior to 1 May 2004
EU-27	The 27 countries that currently make up the EU
FIR	Finite impulse response, a type of digital filter [262]
GDP	Gross domestic product
GIS	Geographic information system
GIScience	Geographic information science (also GISc or GISci)
GNU	GNU's Not Unix
GoF	*Gang of Four*, the authors of *Design Patterns* [198]
GPL	GNU General Public License
GPS	Global Positioning System
GUI	Graphical user interface
GWAP	Game with a purpose [559]
HCI	Human–computer interaction
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
I/O	Input/Output
ICT	Information and communications technology
IDE	Integrated development environment
IEC	International Electrotechnical Commission
IIR	Infinite impulse response, a type of digital filter [262]
IoT	Internet of Things (see section 3.2)
IP	Internet Protocol
IPC	Inter-process communication
ISO	International Organization for Standardization

IT	Information technology
Java EE	Java Platform, Enterprise Edition – formerly J2EE (see section E.3)
Java ME	Java Platform, Micro Edition – formerly J2ME (see section E.3)
Java SE	Java Platform, Standard Edition – formerly J2SE (see section E.3)
JCP	Java Community Process [518]
JIT	Just-in-time
JLS	Java Language Specification [**Gosling1996**, 228, 229]
JNI	Java Native Interface [319]
JRC	Joint Research Centre of the EC
JSON	JavaScript Object Notation [104]
JSR	Java Specification Request
LCD	Liquid crystal display
LGPL	GNU Lesser General Public License
LHS	Left-hand side
LPCM	Linear PCM (see section A.4.2.1)
LWUIT	Lightweight UI Toolkit [516]
MID	Mobile information device
MIDlet	An MIDP app – portmanteau of MID and applet (see section E.3.2)
MIDP	Mobile Information Device (MID) Profile [517] (see section E.3.2)
MMAPI	Mobile Media API [283] (see section E.3.3)
MMS	Multimedia Messaging Service
MNT	Mobile Noise Tagger, the mobile part of the NoiseTube Prototype (see appendix C)
MP3	MPEG-1/2 Audio Layer III, an audio encoding format
MP4	MPEG-4 Part 14, a multimedia container format
MPEG	Moving Picture Experts Group
MVC	Model-View-Controller pattern [446]
NGO	Non-governmental organisation
NIHL	Noise-induced hearing loss
OECD	Organisation for Economic Co-operation and Development
OHA	Open Handset Alliance [397]
ORM	Object-relational mapping
OS	Operating system
OSM	Open Street Map [399]
OSS	Open source software
PC	Personal computer

PCM	Pulse-code modulation (see section A.4.2.1)
PDA	Personal digital assistant
PGIS	Participatory GIS (see section 2.4.3.3.2)
PIM	Personal information manager/management
PSEM	Personal sound exposure meter (see section A.5.2)
RAM	Random-access memory
RE	Runtime environment
RHS	Right-hand side
RMS	Root-mean-square
SAX	Simple API for XML
SDK	Software development kit
SLM	Sound level meter (see section A.5.1)
SMS	Short Message Service
spec	Specification
SPL	Sound pressure level (see section A.2.3)
SQL	Structured Query Language
TV	Television
UCS	Universal character set
UI	User interface
UML	Unified Modeling Language [191]
UN	United Nations
URL	Uniform resource locator
USB	Universal Serial Bus
UTF-8	UCS transformation format – 8-bit, a Unicode character encoding
VBR	Variable bit rate
VGI	Volunteered geographic information (see section 2.4.3.3.2)
VM	Virtual machine
VP	Vice-President
WAVE	Waveform Audio File Format, a container format [595]
WCED	World Commission on Environment and Development [602]
WG-AEN	EC Working Group on the Assessment of Exposure to Noise [159]
WHO	World Health Organisation
WSN	Wireless sensor network
WYSIWYG	What you see is what you get
XML	eXtensible Markup Language

Chapter 1

Introduction

1.1 Research context

The need to avert climate change and its perilous effects, the need to stop biodiversity loss caused by human activities, the dilemma of having to feed an ever larger population while trying to preserve the planet's rainforests, getting by with increasingly scarce supply of fossil fuels and other resources, and coping with mounting environmental stressors and social pressures threatening health and quality of life in rapidly expanding agglomerations – these are just a few of the daunting sustainability problems humanity faces today. To safeguard our well-being and that of future generations we must urgently move towards sustainable development. This means we must find ways to reconcile demographic and economic growth with the protection and restoration of the environment and the preservation and improvement of the quality of human lives everywhere.

This challenge should be a concern to literally everyone, from policymakers to citizens. But when it comes to finding answers most eyes turn to science and technology. Hence, scientists of all disciplines have a responsibility to ask themselves what their field can do to contribute to the search for solutions and their implementation in society.

Driven by the ambition to make a meaningful contribution to this challenge, the research covered in this dissertation investigates how the latest developments within computer science and ICT can be applied to establish tools and practices that allow us to better understand, manage and ultimately protect our environment, and thereby our quality of life. Our research is part of the *citizen science* movement. From a computer science perspective it can be situated in the field of *mobile sensing*, which is itself rooted in the wider movement of *ubiquitous / pervasive computing* and that of the *Internet of Things*.

1.2 Problem statement

Most of the sustainability problems humanity faces on global and local scales can be seen as examples of the overexploitation of a *commons*. As we will explain in chapter 2, there is a growing consensus among social scientists and policymakers about the fact that sustainable exploitation of commons, and the tackling of environmental issues in general, requires broad participation and awareness of the general public. Citizen communities should be allowed to participate in the process of environmental monitoring and the establishment of new institutions, whether these are ad-hoc rules created within communities or laws passed by authorities, that aim to protect the commons they rely on.

Therefore the problem statement of this thesis is:

How can contemporary ICT be applied to establish participatory, low-cost tools and practices that enable communities to monitor, raise awareness about, and sustainably manage the commons they rely on?

Box 1.1: Problem statement of this thesis

1.3 Research goals

The work presented in this dissertation serves three principal goals:

Goal 1: Formulate a general approach
It is our ambition to formulate a general approach that constitutes an answer to the problem statement. This approach should be abstract enough to transcend the context of specific communities facing specific commons issues, but at the same time concrete enough to serve as a blueprint for solutions that can be implemented and deployed in practice to help specific communities deal with specific problems.

Goal 2: Apply it to a specific, socially relevant case
We intend to conduct applied, interdisciplinary research aimed at turning our general approach into a concrete, ICT-based solution for a specific, socially relevant problem. To design and implement this solution we must apply state of the art technology and make advances where necessary. This effort is interdisciplinary because it requires us to gather domain knowledge relevant to the chosen case, in order to understand the needs of potential users and the expectations of domain experts. We intend to deploy and validate this solution in real-world conditions to prepare for operational – as in, non-experimental – usage in the relatively short term.

Goal 3: Aim for broad societal impact

Driven by a sense of urgency, regarding the sustainability challenges humanity faces, it is our ambition to conduct research that has real societal impact early on. This way we intend to contribute to the raising of public, academic and governmental awareness about the specific problem we focus on, as well as sustainability challenges in general. This influences the way we conduct research, as well as how, where and to whom we communicate about it.

1.4 Research plan

Before we can formulate a general approach (goal 1) we must first develop a vision regarding the way contemporary ICT can support communities in commons management, and how it can help them to interact with other stakeholders. This vision should cover both technical matters and organisational aspects. The core elements of the vision we introduce are *community memories*, *mobile sensing* and *social tagging*. A community memory is an ICT resource that empowers citizens by enabling them to archive, discuss, visualise and share information that is relevant to the management of a commons they are concerned with. A community can also use this platform to interact with other stakeholders such as authorities, firms and scientists. Through mobile sensing and social tagging citizens can play an active role in environmental monitoring by collecting quantitative and qualitative data using off-the-shelf mobile phones.

Because mobile sensing is the subject of active research, it is necessary to have insight in the current state of the art and open challenges. This allows us to distil a set of specifications for mobile sensing systems aimed at environmental monitoring in a community memory context. Moreover, it allows us to pin down and prioritise the open challenges which we need to tackle in order to realise our vision.

The concrete problem we address (goal 2) is that of *environmental noise*, commonly referred to as *noise pollution*. Environmental noise is the noise people are exposed to in their daily lives as a result of various human activities, such as those related to transport, industry and leisure. Long-term exposure to excessive environmental noise affects human behaviour, productivity, health and general quality of life. Due to economic and industrial development environmental noise is becoming an ever greater concern in agglomerations around the world. We have three main motivations for choosing this topic. First and foremost, it is a real, pressing issue which affects millions of people every day. Second, due to recent European legislation there is a major incentive to develop new methods to assess environmental noise. Third, unlike some other environmental problems such as air pollution, noise pollution can be measured using mobile phones without any additional, potentially expensive, measuring equipment, as we demonstrate in this work.

In order to be able to propose a solution that is applicable in practice, it is essential that we understand the needs of potential users – i.e. noise-concerned citizens – and the expectations of domain experts – i.e. acousticians, policymakers, officials of environmental agencies, etc. Therefore we need to assemble the necessary expertise regarding acoustics, human hearing, subjective aspects of sound perception as well as current governmental policies and conventional methods for the assessment of environmental noise. This is one of the aspects which make this an interdisciplinary effort.

We apply this expertise to propose a novel, participatory solution to the assessment of environmental noise and the perception of (urban) soundscapes in general. This solution is enabled by *NoiseTube*, a mobile sensing and community memory system that allows citizens to contribute to the assessment of environmental noise and its impact on the quality of life of local communities. Rather than only develop the technology we also experiment with different deployment and coordination strategies, and reach out to potential adopters (e.g. individual citizens, NGOs and local authorities).

We allow anyone to use the NoiseTube system for free and we make publicity for it by presenting our work at various events and by reaching out to mainstream media. This way we intend to convince as many people as possible to try out NoiseTube in order to help us improve it, while contributing to the creation of noise maps for their local area. Moreover, we seek to get the attention of NGOs or local authorities with whom we can forge partnerships to set up coordinated noise mapping efforts using NoiseTube. With these efforts we also directly contribute to the raising of awareness about the problem of environmental noise (goal 3).

The quality of data collected by citizens using devices that are not purpose-made is bound to be disputed. Hence it is crucial that we validate our solution, especially with regards to the credibility of the sound level measurements made on mobile phones and the general results of participatory noise mapping efforts. Therefore we make significant efforts to conduct validation experiments in the lab and the field. Laboratory experiments allow us to evaluate and improve the accuracy of the sound level measurements the NoiseTube system allows to collect. For the field experiments we collaborate with a group of citizens and coach them to use the NoiseTube system to assess noise levels in their neighbourhood. With these real-world experiments we evaluate the usability of the system, the effectiveness of collaboration strategies, the quality of the aggregated data and the suitability of the noise maps that we generate from it.

1.5 Structure of the dissertation

Here we summarize each subsequent chapter of the dissertation:

Chapter 2: Context & vision

In this chapter we develop the rationale behind our research. We sketch the challenge of sustainable development and distil guidelines from theory and practice regarding commons management and environmental policy in general. We outline the opportunities which today enable us to tackle these problems in a new, participatory way. Building on these guidelines and opportunities, we present our vision of community memories as platforms for participatory environmental monitoring and governance of commons.

Chapter 3: Mobile sensing

On the one hand, this chapter serves to situate our work within the field of mobile sensing and the wider movement of ubiquitous computing research. To this end, we clarify the origins, current state and open challenges of the field of mobile sensing. On the other, it concretises our approach by contrasting it with the work of others and by identifying the specific challenges we have to tackle to develop mobile sensing systems for environmental monitoring in a community memory context.

Chapter 4: All about noise

In this chapter our goal is to shed light on two sides of the concept of noise. First, we study definitions of noise at the level of individual sounds and individual hearers. Then we focus on environmental noise as a societal problem and discuss its effects on human health and well-being, the response of policymakers, and current assessment methods.

Chapter 5: The NoiseTube system

This chapter introduces our participatory approach to environmental noise assessment and the NoiseTube system which underpins that approach. The NoiseTube system consists of two principal components: the *NoiseTube Community Memory* and the *NoiseTube Mobile* application for smartphones. We treat the functionalities and implementation aspects of the NoiseTube Community Memory – leaving the mobile application for the next chapter. In this chapter we also provide statistics about when, where, how and by whom the system has been used so far. We also discuss our experience with the early deployment of the system.

Chapter 6: The NoiseTube Mobile app

In this chapter we take a detailed look at the functionality, design and implementation of NoiseTube Mobile. This freely downloadable participatory sensing app turns a mobile phone into a personal, portable, low-cost sound level meter. Additionally this chapter also discusses the main examples of related work – with respect to NoiseTube Mobile and the NoiseTube system as a whole.

Chapter 7: Validation

In this chapter we discuss our efforts to validate the NoiseTube approach and system – in terms of suitability and usability – by means of experiments in lab and the field. The main challenge we tackle here is that of evaluating and improving data quality, both at the level of measurements made by individual mobile phones and at the level of coordinated noise mapping campaigns involving groups of citizens. This validation work is essential given our ambition to make participatory noise mapping an acceptable alternative or complement to conventional assessment methods.

Chapter 8: Conclusion

To wrap up the dissertation this chapter lists the main contributions of our work and provides an overview of on-going and future work.

Finally this dissertation contains seven appendices:

Appendix A: All about sound

This appendix provides further background information to support the comprehension of chapters 4 to 7. It introduces the basics of acoustics, human hearing, analogue and digital audio signals, and of sound level meters – the devices whose functionality the NoiseTube Mobile app brings to the platform of mobile phones.

Appendix B: Precision, accuracy & calibration

This appendix provides a short recap of high school physics on the distinction between precision and accuracy, different types of measurement errors and how they can be remedied by averaging or calibration.

Appendix C: The NoiseTube Prototype

Here we briefly discuss and early prototype for the NoiseTube system, developed in the summer of 2008.

Appendix D: NoiseTube data interchange specifications

This appendix documents the data interchange APIs and file formats that handle the communication and exchange of data between NoiseTube Mobile and the NoiseTube Community Memory.

Appendix E: All about platforms

This appendix provides background information on the hardware (i.e. smartphones) and software (e.g. Java ME and Android) platforms we have targeted with NoiseTube Mobile. Apart from describing technical aspects, we also spend attention on the evolutions in the market for smartphones and the software platforms that run on them. Moreover, based on our experiences we also present generalised guidelines regarding cross-platform application development for Java ME and Android.

Appendix F: Questionnaire

In order to evaluate the NoiseTube system from the perspective of end-users we have created a questionnaire which is included in this appendix.

Appendix G: Dissemination & impact

Here we give an overview of the dissemination of our work and the impact it has and continues to have. This includes scientific and other publications, contributions at events, and media mentions.

Chapter 2

Context & vision

2.1 Introduction

In this chapter we develop the rationale behind the research presented in this dissertation. At its core, our research is focused on the creation of low-cost ICT systems, and associated practices, to enable ordinary citizens to participate actively in the process of environmental monitoring and governance. By leveraging mobile and Web technology, we aim to raise public awareness of, and facilitate innovative, participatory solutions for, local environmental and quality of life issues, and sustainability challenges in general.

In order to contextualise, motivate and underpin our vision and approach, this chapter frames our research in a broader societal context and draws links with diverse sources of inspiration and related work within and beyond computer science. First, section 2.2 sketches the challenge of achieving sustainable development on global and local scales. Next, in section 2.3 we distil guidelines from research and practice regarding the management of commons and environmental issues in general. Then, in section 2.4 we outline three principal opportunities – in terms of people, technology and practices – which today enable us to tackle these problems in a new, participatory way. Building on these guidelines and opportunities, we present our vision on participatory environmental monitoring and governance of commons in section 2.5. Section 2.6 concludes this chapter.

2.2 The sustainability challenge

Sustainability, in its many facets, is arguably the main challenge faced by humanity today, both on global and local scales. We believe scientists and technologists of all disciplines bear a responsibility to do what they can in this matter.

2.2.1 Global issues

On 31 October 2011 the world population hit the 7 billion milestone. By the same UN projections, it is expected to exceed 10 billion well before the end of this century [538]. An increasing amount of evidence indicates that this demographic growth, in combination with increasing economic development and rising socio-economical aspirations worldwide, imposes unsustainable demands on our planet's resources.

It is estimated that humanity's combined ecological footprint has exceeded the Earth's carrying capacity since the late 1970s or early '80s, and has continued to grow ever since [343, 560]. In 2007, we collectively lived as if we had 1.5 Earth-like planets at our disposal, and, if the trend continues, it is expected that by 2030 we would need 2 such planets to sustain our demands [426]. The consequences of this overexploitation are ecological as well as humanitarian, moreover, often the former leads to the latter. For instance, human-caused decline in biodiversity[1] not only represents a loss of wildlife[2], which is tragic in itself, but also risks to impede, among other things, food provision (e.g. due to depletion of fisheries[3] and crop pollination problems), freshwater supply[4], wastewater treatment, medicine provision and the absorbance of greenhouse gases by threatened ecosystems such as rainforests [426].

That brings us to the topic of climate change, arguably the most prominent sustainability challenge today. In its latest assessment report, the UN's Intergovernmental Panel on Climate Change stated that the warming of the climate is considered to be an unequivocal fact, and that most of the increase in global average temperature is « *very likely* » caused by increased greenhouse gas concentrations due to human activity [276]. Here too, the warming of the climate in itself is not the main problem, but the secondary consequences are truly worrying. Examples are the increase of drought in dry regions (which are also expected to become larger due to desertification), the increase of precipitation and thereby flood risk in wet regions, further losses of biodiversity due to overtaxed resilience of

[1] The *Living Planet Index*, the main biodiversity indicator, has dropped by ±30% since 1970 [426].

[2] A single yet striking example is that there are now more tigers living in captivity than in the wild [426].

[3] In 2008, an est. 83% of the world's oceanic fisheries were fished at or beyond capacity [188: p. 38].

[4] In 2010, 1.8 billion people used the Internet, yet 1 billion lacked adequate freshwater supply [426].

ecosystems, increases of coastal erosion and flooding due to sea-level rise and extreme weather events [73, 276].

Another sustainability challenge is energy provision. Fossil fuels are the main source of consumed energy worldwide [545] and in the developed world they account for about 80% [330]. The dependency on fossil fuels is unsustainable for at least three reasons. First, fossil fuel reserves that can be exploited in an economically viable way will run out at some point – although estimates of when this will happen vary considerably. Long before that, decreasing supply and increasing demand due to global economic development, may well cause energy prices to skyrocket. Second, fossil fuel-based energy provision causes 74% of worldwide greenhouse gas emissions and is thus (*very likely* [276]) a primary contributor to climate change. Third, fossil fuel-related interests – concerning both exploitation of reserves and transportation of extracted fuel – have already caused or contributed to numerous episodes of (geo)political instability, including outright war, in various places around the world. As reserves run lower and prices go up, the risk of conflicts will probably only increase. Unless fossil fuel dependency is reduced, *energy security* is therefore likely to become an increasingly challenging concern, especially for countries that have no reserves of their own or have already (largely) depleted them [330].

Finding solutions to global sustainability issues requires profound thinking about all forms of human development and the limits that govern it. A pioneering work on the topic is *The Limits to Growth* from 1972 [342]. This report, commissioned by the Club of Rome think tank, investigated the consequences of the rapidly growing world population (then approaching 4 billion) in relation to the Earth's finite resources. It put forward a model which showed that world population, industrialisation, pollution and resource depletion were all on a path of exponential growth, while technological advances could bring about only linear increases in availability of resources. The report stirred considerable debate, yet many critics dismissed it as a doomsday prophecy [388]. However, the actual global development trends observed since 1972 have been largely in agreement with the predicted growth scenario [343, 532]. This, combined with the mounting scientific evidence on the above-mentioned and other problems related to humanity's growing ecological footprint, makes it all the more likely that we, as a species, are about to reach, or have already surpassed, the limits to growth.

Yet global economic growth and development are also seen as principal drivers for the amelioration of human lives everywhere. The true challenge humanity faces is therefore a balancing act between sustainability and development. In 1987, *Our Common Future*[5], a report by the UN's World Commission on Environment and Development (WCED), defined *sustainable development* as « *development that meets the needs of the present without compromising the ability of future generations to meet their own needs* » [602].

[5] Also known as the *Brundtland Report*, after former Norwegian Prime Minister Gro Harlem Brundtland, who chaired the WCED – also known as the Brundtland Commission.

2.2.2 Local issues

Also at local scales we see growth processes that may be unsustainable in the long run. An example is the process of urbanisation. In 2005, the UN estimated that cities were home to 49% of the world population, and predicted a rise to 60% by 2030 [537]. In 2008 the European Commission's Joint Research Centre and the World Bank estimated that 95% of the world population was concentrated on just 10% of the planet's land area [368]. Urbanisation is especially rampant in the developing world: according to the UN, 74% of the world's city dwellers lived in developing countries in 2005, and by 2030 this is expected to be 81% [537]. While urbanisation has positive effects as well, the rapid growth of many cities and agglomerations around the world represents an alarming threat to the quality of life of their citizens. This is due to poor sanitation and health services, shortage of adequate housing, unemployment, poverty and other forms of social injustice, crime, transportation problems, lack of (green) space and environmental degradation caused by air, soil, water and noise pollution [539: pp. 240–269].

Global and local sustainability issues are often interrelated. On the one hand, global or transnational issues put mounting pressure on local communities in many parts of the world, for instance due to collapsing fisheries, drought, natural disasters and pollution of "mobile" resources such as air and water. On the other, communities in urbanised areas often have a disproportionately large environmental footprint[6].

2.2.3 A call to arms

Today, given pressing issues such as climate change, achieving sustainable development is more important than ever. This concerns, and requires efforts of, literally everyone, from policymakers to citizens. Yet when it comes to finding answers, most eyes turn to science and technology. Therefore, scientists and technologists of all disciplines should take up the challenge and ask themselves what their field can do to contribute to the search for solutions, both theoretical and practical, *and* their implementation in society.

From a computer science perspective, there are, broadly speaking, two avenues to follow. On the one hand, there is the *green computing* or *green IT* [358] movement, that aims to lower the environmental impact of computing infrastructure itself, typically by increasing energy efficiency through optimisations in hard- and software. On the other, computer science can contribute to the creation of systems and practices that allow us to better understand, manage and ultimately protect our environment, and thereby our quality of life. The work covered in this dissertation is situated in the second category.

[6] It has been reported that cities are responsible for approximately 75 to 80% of worldwide greenhouse gas emissions, although there is considerable debate on the role of other factors besides urban density [134].

2.3 Guidelines

In this section, we present a number of guidelines that have been put forward to cope with sustainability challenges, and which have served as a source of inspiration for our approach to environmental monitoring and governance. We take two perspectives to distil these guidelines. On the one hand, we survey findings of economists and other social scientists in the field of commons management. On the other, we look at common practices applied by policymakers in their efforts to tackle environmental problems.

2.3.1 Commons management

Many of the sustainability problems humanity is facing today, on global and local scales, can be seen as examples of the overexploitation of a *commons* – i.e. a resource which is accessible to all and owned by no-one[7]. This is a theme that has occupied thinkers for at least two millennia [456]. In 1968, biologist Garrett Hardin provided it with the catchy title *The Tragedy of the Commons*, in an eponymous article published in *Science* [245]. This article became very influential, but was also criticised for the remorseless and pessimistic vision it proclaimed. One of the most prominent critics is economist Elinor Ostrom, whose research has shown that self-managed commons need not always end in tragedy. In 2009 Ostrom was awarded the Nobel Prize in Economics [456]. Her findings are an important inspiration for our approach to environmental monitoring and governance.

2.3.1.1 The tragedy

In his famous article, Hardin argued that users of a commons are trapped in an inevitable process that leads to the destruction of the very resource upon which they depend [408]. He argued that the "rational" maximisation of individual benefits by each user of a commons, ignorant of costs imposed on others, cumulates to an overexploitation and a likely eventual collapse of the resource. Hardin illustrated this *tragedy* using the metaphor of a pasture « *open to all* », used by multiple independent herders to graze their herds. He reasoned that each individual herder, motivated by self-interest – or blind greed, if you will – is compelled to increase the size of his herd without limit. Hardin therefore concluded that, in a stable society, where the numbers of both man and beast are not curtailed by warfare, poaching or disease, the pasture would become overgrazed, causing it to degrade and eventually be rendered useless, resulting in ruin for all [245].

[7] Or everyone.

2.3.1.2 Common-pool resources

Hardin's metaphorical pasture is an example of what economists call a *common-pool resource* (CPR). CPRs are defined as resources to which more than one party has access, because exclusion through physical or institutional[8] means is impossible or too costly, and for which exploitation by one beneficiary reduces availability[9] to others. This *difficulty of exclusion* and *subtractability* can lead to dilemmas in which CPR users, called *appropriators*, by acting upon their own short-term interests[10], produce outcomes that are not in anyone's long-term interest. Note that appropriators are not necessarily each other's peers, like Hardin's herders are. For instance, the CPR could be a patch of rainforest appropriated by both indigenous tribes and logging companies. Besides pastures and forests, CPRs include fishing grounds, groundwater basins, or on a larger scale, the Earth's atmosphere and oceans[11]. Apart from natural resources, CPRs can also be products of civilisation, like irrigation systems, the Internet or the Web [315, 408].

2.3.1.3 Conventional solutions

Conventional economic theory has proposed two primary solutions to the CPR problem. Both rely on the introduction of property rights. On the one hand, scholars and policymakers have used Hardin's original statement as an argument for the *centralisation* of all CPRs under government control [408]. In such cases a state takes ownership of the CPR, sets rules concerning access rights and limitations, and typically taxes appropriators to generate funds that can be applied to pay for monitoring[12] and rule enforcement. On the other hand, liberals have argued that all CPRs should be *privatised*. In such cases a single party, usually organised as a private firm, takes ownership of the CPR (often by buying it from a state that took ownership with the intention to privatise it), enjoys exclusive access to it and typically carries out, or at least finances, monitoring to keep out others. In both cases, at least part of the original appropriators are disenfranchised. Moreover, the introduction of property rights, regardless of whether they are held by a central state or a private firm, arguably means that CPRs are no longer truly "common".

---

[8] *Institutions* are sets of rules that govern human interaction [456: p. 1].

[9] In terms of accessibility, quantity and/or quality.

[10] In absence of effective rules, limiting access and defining rights and duties, appropriators are likely to *free-ride*, either by overusing without concern for negative effects on others, or by neglecting to contribute to the maintenance of the resource itself (e.g. fertilising the pasture) [408]. In extreme cases appropriators may even resort to violence or other forms of abuse to settle disputes or assert their dominance (e.g. to gain exclusive/privileged access to, or take de facto ownership of, the CPR) [404].

[11] Of which Hardin has said: « *No one owns the Earth's atmosphere. Therefore, it is treated as a common dump into which everyone may discharge wastes. Among the unwanted consequences of this behaviour are acid rain, the greenhouse effect, and the erosion of the Earth's protective ozone layer. Industries and even nations are apt to regard the cleansing of industrial discharges as prohibitively expensive. The oceans are also treated as a common dump.* » [247].

[12] Of the state of the resource itself and of appropriator behaviour.

In 1998, Hardin himself refined his original statement along those same lines [246]:

> « *A 'managed commons' describes either socialism or the privatism of free enterprise. Either one may work; either one may fail: 'The devil is in the details.' But with an unmanaged commons, you can forget about the devil: As overuse of resources reduces carrying capacity, ruin is inevitable.* »

2.3.1.4 A third way

Since the 1970s, a group of social scientists, headed by Elinor Ostrom, has challenged this remorseless logic by advocating a third, previously discarded, solution in which CPRs are retained as common property and appropriators are left to set up their own system of collective governance. Ostrom and her peers argue that, under certain conditions, this "third way" *can* lead to sustainable outcomes, even there where conventional solutions may fail or are simply infeasible [456]. In her seminal book *Governing the Commons: The Evolution of Institutions for Collective Action*, Ostrom outlines the results she and her team had achieved up to 1990 by means of theoretical as well as empirical research [404].

On the theoretical front, Ostrom for instance argues that the tragic outcome of Hardin's pasture metaphor is, at least in part, due to the presumption that the herders will not or cannot communicate, let alone negotiate, amongst themselves to achieve a favourable outcome for all[13]. Regarding the conventional solutions of centralisation and privatisation, she points out that, although they share the central assumption that institutional change must be imposed by an external actor, they cannot both be correct in general because the changes they recommend are contradictory. She therefore warns against models with sweeping, one-size-fits-all claims, and instead recommends that different CPR problems be tackled with tailored institutional solutions, which take time and effort to establish. As an alternative solution to the herders dilemma, Ostrom presents an arrangement in which the herders themselves cooperate to reach and enforce a binding contract to exploit the common pasture sustainably [404: pp. 2–18].

On the empirical front, Ostrom and her team have collected and analysed data on actual CPRs that are collectively managed, either successfully or unsuccessfully. All over the world and across various economic sectors, they found examples of CPRs that were (and are) indeed being sustainably exploited through collective arrangements set up by the appropriators themselves, without resorting to centrally or privately held property rights. This empirical work was distilled into a set of design principles, which we list in box 2.1. These help to account for the success of long-enduring, self-organised institutions for CPR exploitation [404: pp. 18–21 & 58–102]. It is noteworthy that principles 4 and 5 challenge the conventional notion that rule enforcement is best left to impartial outsiders [456: p. 11]. In *Governing the Commons* and later publications Ostrom documents

[13] This has led others to formalise the metaphor as a *prisoner's dilemma* (cf. game theory) [402: pp. 12–16].

these principles in detail and gives explanations as to why they contribute to desirable outcomes. Although these principles do not provide an easy solutions to the often complex CPR problems, in cases where they are all heeded « *collective action and monitoring problems tend to be solved in a reinforcing manner* » [405: p. 267, 456: p. 12].

1. **Clearly defined boundaries**
   Rules should clearly define who has what entitlement. Individuals or households who have rights to withdraw resource units from the CPR must be clearly defined, as must the boundaries of the CPR itself.
2. **Congruence between appropriation and provision rules and local conditions**
   Appropriation rules restricting time, place, technology, and/or quantity of resource units are related to local conditions and to provision rules requiring labour, material, and/or money. An individual's duty to maintain the resource should stand in reasonable proportion to the benefits.
3. **Collective-choice arrangements**
   Governance is more successful when decision making processes are democratic: the majority of individuals affected by the operational rules can participate in modifying them.
4. **Monitoring**
   Active monitoring of CPR conditions and the behaviour of appropriators should be carried out either by the appropriators themselves or by someone who is accountable to them.
5. **Graduated sanctions**
   Appropriators who violate operational rules should be subjected to gradual sanctions by other appropriators, or by officials accountable to these appropriators, or by both. The gradation of sanctioning should depend on recidivism, the seriousness and context of the offence.
6. **Conflict-resolution mechanisms**
   Appropriators and their officials should have rapid access to low-cost local arenas to resolve conflicts adequately among appropriators or between appropriators and officials.
7. **Minimal recognition of rights to organise**
   The rights of appropriators to devise their own institutions are not challenged by external governmental authorities.

For CPRs that are parts of larger systems:

8. **Nested enterprises**
   Appropriation, provision, monitoring, enforcement, conflict resolution, and governance activities are organized in multiple layers of nested enterprises.

Box 2.1: Design principles for long-enduring CPR institutions as observed by Elinor Ostrom [404: p. 90, 456: p. 11]

As noted above, Ostrom is a strong proponent of *institutional diversity*. She argues that different CPR problems require different, tailored institutional solutions, because aspects of resources and appropriators may determine which governance regimes are feasible and which issues may arise during deployment. Determining resource characteristics are size, measurability, carrying and storage capacity, temporal and spatial availability, regeneration rate and whether it is stationary (e.g. trees in a forest) or mobile (e.g. migratory fish), all of which may for instance affect the feasibility and cost of monitoring. As for appropriators themselves, cultural differences as well as the size of the group may be determining factors, because they affect organisational complexity and attributes of human relations such as reciprocity, trust and reputation (e.g. individuals that know or can identify one another are more likely to develop shared norms than total strangers) [408]. Ostrom et al. indicate that a lack of knowledge of local conditions, like those listed above, can undermine the legitimacy of governmentally imposed restrictions – as in centralisation strategies – making them counterproductive and prone to tragic failure [128, 456: p. 10].

Ostrom notes that technological advances – such as wireless communication systems, sensors, tracking devices, geographical information systems and the Internet – enable more effective management of CPRs by larger groups of people. However, she also stresses that, while technology can help to inform decisions, it is *not* a substitute for decision-making [408]. This notion is of great importance to this thesis.

In recent work, Ostrom and co-authors have reflected on if and how theory and practice of successful commons management, typically concerning resources managed by relatively small groups of people within a single country, can be applied to pressing commons challenges that span multiple countries or even the entire globe, such as fisheries and fossil fuel reserves in international waters and global atmosphere and climate. They note that these challenges are significantly harder to tackle, since they require international cooperation and are further complicated because, at that level, resources may be intrinsically difficult to measure (due to extreme size and complexity) and CPR problems are often interlinked (e.g. climate change and biodiversity loss). They argue that, while some experience from smaller CPR systems may transfer directly to larger ones, finding appropriate, adaptive institutional solutions for global commons problems will require additional multi-scale and multidisciplinary research [128, 406–408].

Elinor Ostrom was made the co-laureate of the 2009 Nobel Prize in Economics, in recognition of « *her analysis of economic governance, especially the commons* » [456].

2.3.2 Policy practices

Another source of inspiration for our vision and approach are common practices found in governmental environmental policies. When looking at the way authorities are addressing, or intend to address, sustainability challenges – including the management of (centralised) CPRs – we notice that public participation and awareness on the one hand, and intensive monitoring and data collection on the other, are considered more and more important.

2.3.2.1 Public participation & awareness

In recent decades policymakers at various levels, from international bodies to local authorities, have stated that public participation[14] and awareness is of great, if not vital, importance to the success of any policy dealing with environmental and sustainability challenges. A principal argument for raising public awareness of environmental problems is that it can help to increase the legitimacy of the often unpopular measures taken by authorities to tackle such issues. A related argument, both for participation and awareness, is that tackling these problems often requires profound changes in citizens' behaviour, which can be difficult or impossible to impose through legislation alone.

An exemplary statement in this matter is *Principle 10* of the *Rio Declaration* [540], proclaimed at the 1992 United Nations Conference on Environment and Development, which reads as follows:

> « *Environmental issues are best handled with participation of all concerned citizens, at the relevant level. At the national level, each individual shall have appropriate access to information concerning the environment that is held by public authorities, including information on hazardous materials and activities in their communities, and the opportunity to participate in decision-making processes. States shall facilitate and encourage public awareness and participation by making information widely available. Effective access to judicial and administrative proceedings, including redress and remedy, shall be provided.* »

2.3.2.2 Monitoring & data collection

Another trend in environmental policy across various levels is a growing ambition to encourage and invest in monitoring of ecological systems and the collection of data on the environmental and social impact of human endeavours *and* citizens' opinion thereof. As we learned above, monitoring and gathering knowledge about local conditions, are essential components of any model for CPR management, including centralised schemes. Also from a more practical point of view, there are solid arguments for authorities to engage in environmental monitoring and data collection:

- Monitoring is indispensable to enforce the *internalisation of environmental costs* (i.e. "making the polluter pay"), which is increasingly seen as a powerful – yet arguably insufficient – policy instrument to curb adverse environmental effects of various industries and modern society in general[15];

[14] Obviously "participation" can mean many things. As we see it, what is being aimed for is a level of public involvement that goes, at least, beyond conventional democratic tools such as elections and referenda.

[15] An example of such policies are the *Cap & Trade* schemes for the reduction of CO_2 emissions.

- Data about physical impacts and human perception thereof can advise policymakers, in preventive as well as curative decisions, in the short run;
- Large datasets are essential to increase scientific insight into the causes and effects of environmental problems, and to construct models to predict future outcomes (e.g. climate simulations), both of which can be applied to advise policy decisions in the longer run.

Such intentions were also put forward in the 1992 Rio Declaration [540], as illustrated by *Principle 16*:

> « *National authorities should endeavour to promote the internalization of environmental costs and the use of economic instruments, taking into account the approach that the polluter should, in principle, bear the cost of pollution, with due regard to the public interest and without distorting international trade and investment.* »

As well as *Principle 17*:

> « *Environmental impact assessment, as a national instrument, shall be undertaken for proposed activities that are likely to have a significant adverse impact on the environment and are subject to a decision of a competent national authority.* »

It is noteworthy that, in order to have an impact on public opinion and policymakers, sustainability-concerned NGOs often also engage in monitoring and data collection activities to construct evidence bases on global and local issues. This applies to both large international organisations (e.g. WWF [604]) and local ones, including loosely-organised *grassroots* activism movements. Besides working independently it is not uncommon that NGOs seek the assistance of academics or other professionals, for instance to increase the scientific credibility of collected evidence.

2.3.3 Summary

Moving towards sustainable development requires that we collectively learn how to sustainably manage common-pool resources at local as well as global scales.

Looking at theoretical and empirical research on commons management, we learned that solutions based on collective arrangements should not be dismissed upfront, as they once were. Collectively established institutions *can* lead to sustainable exploitation of CPRs, even where conventional solutions fail. Of paramount importance to reach long-enduring CPR institutions is the participation of a majority of stakeholders in decision making processes, in the monitoring of resources and appropriator behaviour and in the enforcement of rules through gradual sanctioning. Establishing such institutions requires time, effort and knowledge of local conditions. New technologies can facilitate successful commons management, but only when embedded in sound institutional arrangements.

Taking the perspective of policymakers, we learned that the stimulation of participation and awareness of the general public is considered to be of great importance for the establishment of effective environmental policies. Authorities do this because measures taken to tackle environmental problems require legitimacy in the mind of citizens, as well as a certain willingness to act or change behaviour. To execute current environmental policies and to advise decisions on new policies, in the short and longer run, authorities must be able to rely on large, up-to-date datasets on the environmental and social impact of various human endeavours, measured in both objective and subjective terms. Assembling such datasets requires extensive monitoring and data collection campaigns.

Technological as well as organisational limitations can significantly hamper the establishment of environmental policies by authorities, as well as (other) institutions for commons management. On the technological front, effective monitoring of resources and human behaviour on a large (possibly global) scale may be either technically infeasible or prohibitively expensive[16]. On the organisational front, the raising of wide public awareness and the inclusion of a majority of stakeholders in decision making may be prohibitively complicated and/or expensive. Finally, reaching truly well-informed decisions may require more data than is technologically or financially attainable.

During the last few decades ICT has revolutionised virtually all aspects of modern life. It has affected the way we communicate and collaborate, in both private and professional/-public contexts, and it has changed – largely for the better – the way various authorities gather information, take decisions and carry out policies. Therefore, we are convinced computer science has a principal role to play in tackling today's sustainability challenges. In this section we learned that, whether these are tackled by citizen-led – e.g. collectively managed CPRs, NGOs or grassroots activism – or authority-led initiatives – i.e. governmental policies – there is a need for flexible, low-cost approaches to environmental monitoring and governance, that facilitate broad public participation and awareness raising. We believe there is now a large potential, within computer science and ICT in general, to conceptualise, design, implement and deploy novel approaches that meet those requirements. But, we should remember this requires more than purely technological solutions.

[16] Moreover, large-scale monitoring of human behaviour may also be ethically unacceptable.

2.4 Opportunities

A low-cost, participatory approach to large-scale environmental monitoring and governance requires three principal ingredients. First, anything "participatory" obviously requires participants – hence, we need a certain degree of confidence that such people can be found, motivated and mobilised. Second, large-scale, cost-effective monitoring requires an ubiquitous infrastructure of interconnected, affordable sensors. Third, enabling large groups of people to participate actively and effectively requires practices for collaborative creation, annotation, management, dissemination and application of various kinds of content and data. In short, we need a platform of *people*, *technology* and *practices*.

Below we explain that over the course of the last decade these ingredients have become available. Hence, there is now a unique opportunity to create systems for environmental monitoring and governance on a spatio-temporal scale and with a level of participation that was simply unattainable before.

2.4.1 People

Building the technology for a participatory system, here for environmental monitoring, is complicated, but finding and keeping participants to make it actually work may be just as challenging. However, the growing concern about sustainability in all layers of society, especially in developed countries, makes us guardedly optimistic that there is a large number of potential participants for the sort of systems and projects we have in mind.

Recent initiatives such as *An Inconvenient Truth* [232], former US vice-present Al Gore's 2006 documentary film on climate change, complemented by an increased attention for the topic in mainstream media, have almost certainly had an effect on the public opinion regarding climate change, sustainability and the environment in general.

In a communication from June 2011 [156], reflecting on the almost two decades since the *Rio Declaration on Environment and Development* [540] in anticipation of the UN's upcoming *Rio+20* conference [536], the European Commission stated that:

> « *There has* [...] *been a major increase in scientific information and public awareness of environmental issues, in particular climate change, and the participation of civil society in global policy-making, not least thanks to improved internet communication.* »

In Europe, citizens' attitudes towards climate change and the environment are regularly assessed by the *Eurobarometer* surveys. Chart 2.1 shows the level of concern EU citizens

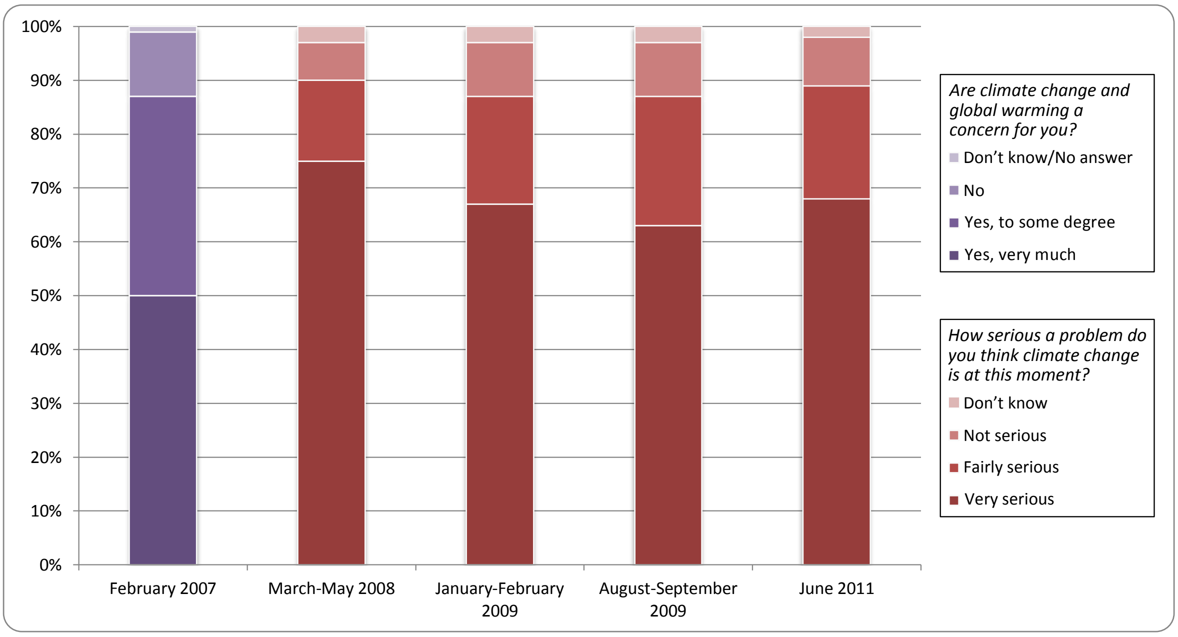


Chart 2.1: EU citizens' concern about climate change in 2007–2011 [158: p. 7, 165: p. 13, 166: p. 16, 162: p. 16, 164: p. 14]

have about the issue of climate change in recent years. Close to 90% consider climate change a very or fairly serious problem [158, 162, 164–166]. Regarding the environment in general, in 2011, 95% of EU citizens consider protecting the environment important to them personally, while 77% believe the state of the environment has an impact on their quality of life (compared to 75% for social and 85% for economic factors). Reassuringly, 87% of EU citizens also believe that the protection of the environment is, at least in part, the responsibility of citizens themselves, and 69% of them believe that citizens are currently not doing enough in that regard [163: p. 7].

Even in the USA, which is one of the few countries in the world that still refuse to ratify the Kyoto protocol on climate change [543], and where to this day many prominent politicians simply deny there even *is* such a thing as global warming [146], a nationwide study by researchers from MIT in 2006 found that 49% of the population considered it to be the most or second most important environmental problem their country faces [108].

Whether public concern translates in a willingness to act or change ways is hard to tell, but it is surely a necessary condition. Finding, motivating and retaining participants is always a major challenge for any participatory system and requires more than purely technological solutions. In case of "environmentally-oriented" systems, we are convinced that concerns about the (local) environment can be a motivating factor[17]. Therefore, we think the current widespread environmental concern indicates that there is a baseline of potential adopters for participatory environmental monitoring.

[17] Our experience with the NoiseTube system has confirmed this (see chapter 7).

It remains to be seen whether sustainability will stay this big a concern for public opinion. Other problems may well (temporarily) overshadow it[18]. Still, we expect that authorities and NGOs will strive to keep environmental issues high on the political agenda and the public mind-set. Especially in Europe, where ambitious commitments were made[19], there is a vested interest to keep the momentum going. Statements such as the above-cited view of the European Commission indicate that new technologies can play a valuable role here. Therefore, supporting the awareness-raising process is also a principal element of our vision for participatory environmental monitoring, as we will explain in section 2.5.

2.4.2 Technology

Building cost-effective systems that enable environmental monitoring and data collection on a wide scale requires a technological platform that is both ubiquitous and affordable. Recently, such a platform has become available in the form of mobile phones and especially *smartphones*, which have caused what has been called a revolution in computing [144].

In a 2008 article in *Communications of the ACM*, Cuff et al. wrote that [106]:

> « [...] *pervasive computing has entered the backpack, purse, and coat pocket in the form of mobile phones, laying the groundwork for Mark Weiser's vision of ubiquitous computing* [566]*.* »

Mobile phones have indeed become truly ubiquitous: the International Telecommunication Union estimates the world now counts 5.9 billion mobile phone subscriptions [275]. Moreover, mobile phones have indeed become miniature computers: so-called *smartphones*[20] have enough processing power to rival fairly recent PCs, connect to the Internet, run user-installable software (i.e. *"apps"*) and come with a multitude of sensors[21] [455].

In recent years smartphones have become increasingly popular. In Western Europe, they are expected to outnumber other mobile phones by 2014 [69], and in some European countries this is already the case [339]. Worldwide, smartphones account for 25% of all new mobile phones sold during the first three quarters of 2011 [202–204][22]. In parallel, mobile broadband[23] is seeing massive uptake, to the point that there are now twice as

[18] For instance, the global economic downturn, which started in late 2008, is blamed for the dip in public concern about climate change in Europe in 2009 (clearly visible in chart 2.1) [162, 166].

[19] Notably regarding the reduction of greenhouse gas emissions, the increased use of renewable energy sources and efforts to increase energy efficiency [168].

[20] Refer to section E.1 for a definition of smartphones and the differences with other mobile phones.

[21] For instance, cameras, microphones, accelerometers, gyroscopes, GPS and other positioning systems, magnetometers, proximity sensors, light sensors and even thermometers and barometers.

[22] Refer to section E.2 for a more detailed discussion of the smartphone market.

[23] This is fast wireless data communication via cellular networks – of 2G and especially 3G or 4G types. It is primarily used to access the Internet.

many mobile than fixed (i.e. wired) broadband subscriptions worldwide [275]. This has enabled smartphones to become a new platform for end-user software and the consumption of digital content (music, video, books, newspapers, maps, etc.), typically delivered through (*app*) *stores*[24].

To some extent these trends represent a democratisation of technology. In recent years, smartphones have become available at lower prices, although high-end devices remain considerably expensive. In developing countries cellular networks are introducing millions of people to the Internet who previously had no (reliable/affordable) access method due to a lack of fixed (broadband) infrastructure [275]. The (mobile) Internet and the app store distribution model have disrupted conventional business models for content and software delivery, thereby enabling cheap or free (yet usually ad-supported) access to digital content and a myriad of personalised, “social” or location-based services.

Scientists have quickly realised that the ubiquity and traceability (from the perspective of network operators) of mobile phones creates an interesting platform for research. For example, anonymised data collected by network operators has been used to study human mobility patterns and the dynamics of cities [46, 212], requiring no phones to be programmed and no subscribers to be even aware of the experiment. However, since it is often hard[25] to obtain data about subscribers from operators, and additional, locally-collected data may be useful or required, similar studies of human mobility, social habits, modes of transport, etc., were carried out with programmed devices [139, 140, 444]. The rise of relatively cheap, Internet-connected, easier to program, GPS-equipped, sensor-laden smartphones, has vastly increased the potential for these and other scientific applications [308, 310, 415, 449]. There are two main insights, both still fairly recent, that have unlocked a wide range of new scientific uses of smartphones.

First is the notion that, besides human behaviour, these devices can be used to study the surroundings in which humans wander [106, 143, 308, 415]. Especially in urban contexts, or wherever people flock together, this enables monitoring and data collection scenarios, for environmental as well as other purposes, on previously unattainable scales and levels of spatio-temporal granularity. Because smartphones are personal devices and are typically carried on or close to the body, they also provide an intimate, *people-centric* perspective on reality, which is difficult to achieve with conventional, dedicated measuring equipment or *wireless sensor networks* (WSNs)[26] [74]. Apart from relying on the phones' built-in sensors, the suite of measurable physical parameters can be extended by attaching external sensors or by wirelessly communicating with nearby sensor-equipped devices.

[24] Such as Apple's iTunes Store [31], Google Play [219] and the Nokia Store [382].

[25] Or even legally impossible.

[26] See chapter 3.

Second is the idea that mobile phone users can be involved in a way that goes beyond being passive (possibly even unaware) vehicles for sensors. Users themselves can act as *human sensors*, adding contextual, semantic or subjective information to augment physical measurements or to contribute to large-scale surveys of public opinion. This *participatory* perspective [71] enables new scenarios in which mobile phone users conduct scientific fieldwork in the spirit of *citizen science*, a practice we discuss below.

To the best of our knowledge, the earliest publications that explored these and similar scientific uses of mobile phones appeared around 2005–2006 [71, 74, 148, 419]. Since about 2008 – which is also when we became involved – the field, known as *mobile sensing*, as well as a few other names, has gained a lot more traction, also outside of academia. Since then, the rise of mobile apps and app stores has increased the potential scale for research applications [310, 349].

In chapter 3 we provide a thorough discussion of the state of the art in mobile sensing. For now, we conclude that (smart)phones, mobile broadband and the app store distribution model represent a new technological potential that can be applied to build dynamic infrastructures for environmental monitoring and data collection at relatively low cost. Due to its democratising aspects, we are convinced this platform is also well suited to raise awareness of environmental issues, to foster public participation in the study and tackling thereof, and to empower citizens and grassroots movements in the surrounding debate.

2.4.3 Practices

The participation of large groups of people in environmental monitoring and governance requires practices for collaborative creation, annotation, management, dissemination and application of various kinds of content and data, and for open debate and discussion among diverse stakeholders. In this section we identify a number of (mostly) recently emerged, societal, cultural, economic or scientific trends, which we consider as building blocks or sources of inspiration. Most of these trends are directly enabled by, or closely related to, the Web or other technological advances. However, rather than to concentrate on enabling technologies, we focus on the enabled practices – for human interaction and expression, (content) creation, gathering and dissemination of (scientific) data or knowledge, and public participation in governance – which often transcend their origins and technological foundations. Many of the trends and practices we touch upon are related, or even overlapping, but we have nevertheless tried to put them in a logical order. First we treat the topic of Web 2.0 and user-generated content. Next, we focus on the notion of crowdsourcing. Then we move to the practice of citizen science. After that we take a brief look at DIY cultures. Finally we discuss the trend of openness in governance.

2.4.3.1 Web 2.0 & user-generated content

The rise, during the previous decade, of what has become known as *Web 2.0*, has led to profound changes in thinking about the creation and management of online content. Although the term is only loosely defined, the primary characteristic of Web 2.0 sites is that they put their users on the forefront as the site's main driving force [360, 400]. Such sites invite visitors to create, update and share content. Usually content authoring is facilitated using tools such as *wikis* or *blogs*. Compared to conventional websites, where visitors take on the role of passive *consumers* of centrally-authored content, Web 2.0 sites take an open-ended approach in which visitors can also act as active *producers* of *user-generated content*, a model which has been dubbed *prosumption* [451]. The most successful example is undoubtedly *YouTube*, other prominent ones – besides the social network sites discussed below – include *Flickr*, *Delicious*, *craigslist* and *Blogger*.

Additionally, some Web 2.0 sites let users edit, classify, reuse or otherwise improve upon works created by others (i.e. their *peers*), and often the whole of the site's content is collectively managed as a *commons*[27] by the community of users. This aspect has been called (*commons-based*) *peer production*, or *social production*, and is also found in open source software (OSS) development [49]. The prototypical example is the *Wikipedia encyclopaedia* [598], others include *OpenStreetMap* [399], *iFixit* and *StackExchange*.

Web 2.0 and associated practices, like prosumption, commons-based peer production, the harnessing of *collective intelligence* and *crowdsourcing* (see below), have attracted the interest of researchers in computing, social science and those in between (e.g. HCI). Studies of Web 2.0's collaborative approaches to content creation and management, people's motivations to voluntary participate and the organisational structures that govern these online communities, have led to many scholarly [9, 48, 49, 68, 363, 390, 400, 451] and popular publications [360, 370, 523, 569]. Many authors have also reflected on the relevancy and applicability of Web 2.0 practices, or the existence of equivalent or encompassing ones, in a wider socio-economical context [48, 49, 259, 451, 523].

One remarkable finding is that, although many Web 2.0 sites could simply not exist without user-generated content, the vast majority of their users – over 99% in case of Wikipedia [370] – do not contribute anything. This is an example of what social scientists call *participation inequality*. In the context of the Web, this is known as the *1% rule* or the *90-9-1 principle*, which is the finding that in many online communities about 90% of users are *lurkers* who never contribute, 9% occasionally contribute, for instance by editing existing content, and only 1% are responsible for almost all (new) content [370].

Studies have shown that publishing personal writings (e.g. blog posts), or contribution to commons-based peer production communities (e.g. Wikipedia), both of which happen

[27] This also means contributors typically do not, or cannot, claim copyright on their work. Instead, contributor rights are often legally protected by a *Creative Commons* [314] or similar content license.

in absence of monetary incentives, are motivated by a wide range of reasons, including intrinsic and self-interested ones. Often people start out with individualistic motivations. For instance, a blogger may be compelled to share his/her opinion, whether or not it is relevant to others, on a topic of personal interest. A Wikipedia contributor may start a new or edit/extend an existing encyclopaedia entry on a topic in which he/she considers him/herself an expert, possibly out of disappointment with what was (not) there. Over time, "1%-type" contributors may find satisfaction and motivation in the collaboration and interaction with others – although the links between individuals remain weak – and in serving the collective goals of the community or the wider public [9, 68, 390, 523].

Clearly, launching a new Web 2.0 site takes more than a good idea and a technically sound implementation. The main challenge is overcoming the *cold-start problem*, which is a typical chicken-or-the-egg paradox: you need users to supply content, but without content your site is uninteresting or useless for all users, including potential contributors. Surviving the cold-start and reaching a *critical mass* of both passive and active users, requires a baseline of interesting content or general utility. But perhaps most importantly, one needs to create the conditions – through community management, incentive creation, etc. – in which contributors find gratification from their work. Given the wide range of Web 2.0-inspired sites already out there, new initiatives also need to fight for the time and attention of the public. Finally, a good dose of luck may be needed as well, because, in spite of the well-known successes, the list of failed Web 2.0 ideas is long[28].

2.4.3.1.1 Social networks

Social networks, also called *social media*, are a particular kind of Web 2.0 services that have revolutionised the way people interact via the Internet. A social network site can be defined as a service which allows individuals to construct a public or semi-public profile within a bounded system, assemble list of other users with whom they share some connection, and browse through the system via their list of connections and those of others [55]. Such a network typically serves as a platform for sharing personal information, thoughts, photos, videos, etc., among connected users, or across the system. Here, the ratio between consumption and production may be more even, because all users produce at least some content by creating a profile page. Some popular examples of social networks are *Facebook*, *Twitter*, *LinkedIn*, *Myspace* and *Google+*.

In recent years such services have seen truly spectacular uptake. For example, in less than 8 years of existence Facebook has attracted 800 million users [410], or about 11.4% of the world population[29]. These services have given the Web, and the Internet in general,

[28] Refer to [301] for some examples.

[29] This massive scale makes social networks an interesting study object for *network science*. For example, recently, the graph of "friend"-relationships on Facebook has been used to test – and confirm – the well-known *six degrees of separation* idea [37, 534].

a personal and social dimension that was seemingly missing before. This has led many people to spend much more time online, also while on the go[30].

The popularity of social networks has caused other services, both new and existing ones, to incorporate "social" features. However, because the market is now dominated by a few big players, and since users are usually not interested in joining a network none of their friends use, a new social network faces an almost insurmountable cold-start problem, unless it can somehow differentiate itself or is backed by a heavyweight such as Google, Apple or Microsoft. Instead of competing with the likes of Facebook & co, most companies now rely on APIs provided by those existing networks to extend their sites with social features, for example to allow users to easily share links or other resources.

One could argue that social networks have filled a void in the Web by creating spaces in which mild forms of exhibitionism (*"Hey people, look at what I'm doing!"*) and voyeurism (*"Let's see what my 'friends' are doing..."*) meet. Moreover, the various communication functions the networks provide make them powerful alternatives to older Internet applications like e-mail and instant messaging. Hence, for a growing group of people these networks have become the primary channels for communication with family, friends and acquaintances. This is especially the case for teenagers, many of whom first discover social networks because they also serve as a platform for online games.

While the "exhibitionism/voyeurism pact", small-scale communication and entertainment may be primary reasons to join them, usages of social networks go well beyond that. Their most disruptive potential arguably lies in the facilitation of mass communication. For instance, the media and the marketing sector have embraced social networks as cheap, yet effective, platforms for dissemination of personalised news and publicity on a massive scale. More interesting, from our perspective, is that the networks increasingly serve as fora for political discussion, awareness raising, protest and grassroots activism. Often such usage is driven by the belief or hope that interactions via virtual networks can contribute to real-world changes. A widely cited, recent example are the popular uprisings in North Africa and the Middle East, which became known as the Arab Spring. While the real extent to which the outcomes of those events were affected by social networks is still hotly debated [209, 145, 44, 299, 87], there is no doubt that embattled, repressive governments fear their potential. Many have therefore resorted to blocking or manipulating Internet access, which generally led to even more public outrage, as illustrated by this slogan spread on social networks during the 2011 uprising in Egypt [145]:

> « *If your government shuts down your Internet,*
> *it's time to shut down your government.* »

[30] The rise of social networks has stimulated the adoption of smartphones and mobile broadband [40].

2.4.3.1.2 Social tagging

Social or *collaborative tagging* is a practice found on many Web 2.0-type websites. The mechanics are straightforward: users can author, upload or select resources (images, pieces of audio, videos, texts, links to other websites, etc.) and associate these with one or more *tags* or *labels*, which are freely chosen single (or contracted) words. Typically resources and associated tags are, at least by default, public – i.e. visible to all others. Each public tagging action[31] contributes to a collective *folksonomy* graph, which serves as a navigational aid for browsing through tagged resources, used by all visitors of the website [79]. When looking at a resource a user can click on one of its associated tags to be taken to an overview of resources that have been tagged the same way, along with co-occurring tags. Often *tag clouds* are used to visualise (typically by varying font sizes) the popularity of tags associated with particular resources or on the website in general.

The pioneer of tagging sites is *Delicious*, a social bookmarking service, started in 2003 as *del.icio.us*, where people tag links to webpages [461]. Since 2004, many other tagging-based sites have appeared and existing sites have adopted the practice. Examples include *Flickr* for photos, *YouTube* for videos and *BibSonomy* for scientific bibliographies. By now, social tagging and similar practices are common across the Web. Placing *hashtags* in Twitter messages (i.e. *tweets*) is a form of social tagging where tags and resources are basically fused together. The "tagging" of people in photos and videos on Facebook is also broadly similar. Another related practice is *geotagging*, in which resources are associated with geographical coordinates – e.g. the location where a photo was taken.

Usually the primary, intended purpose of social tagging is organisational: people tag a resource with objective – e.g. what, who, where or when – or subjective descriptions, such that they and others can easily find, group and navigate that and similar resources[32]. Contrary to what one might expect, the freedom implied by distributed, open-ended tagging does not lead to complete chaos. Studies have shown that over longer periods the distribution of tags associated with similar items stabilises rather quickly [79, 80, 211]. Despite the absence of global coordination it appears that a collective consensus arises and is maintained over time [492][33]. Social tagging thus constitutes a light-weight, non-hierarchical, decentralised way of organising and opening up huge amounts of information and is a bottom-up alternative to top-down, expert-designed ontologies as often practiced in the semantic web community. However, studies have shown that people's motivations to engage in tagging and to choose certain tags go well beyond organisational purposes. Other motivations include general communication, taking ownership or credit, attracting attention, expressing opinion, self-presentation, humour and protest [393, 529].

[31] Also referred to as a *post* and represented by a <user, resource, {tags}> triple [79].

[32] For example, a picture of the Atomium may be tagged with "Atomium", "Waterkeyn", "Brussels", "Belgium", "Expo58", "1958", "iron", "crystal", "monument", "spectacular", etc.

[33] This also makes folksonomies an interesting subject for research in distributed cognition, semiotic dynamics, emergent semantics and network theory [78–80, 492].

2.4.3.2 Crowdsourcing

The practice of *crowdsourcing* applies to the sourcing of tasks traditionally performed by specific, professional or appropriately trained individuals, to a group of people, a community or the general public through an open call. Some or all of the members of such a *crowd* may be amateurs, in the sense that they are not specifically trained, officially qualified or employed for the task at hand. In some cases the crowd works on a solely voluntarily basis (i.e. for free), but there can also be monetary or other incentives. A principal assumption supporting the practice is that because it is based on an open call, it will attract those who are fittest to perform the task [259]. Moreover, it is assumed that crowdsourcing enables large groups of people to perform functions collaboratively that are difficult to automate[34] or expensive to implement otherwise [236, 259]. The term, which is a portmanteau of "crowd" and "outsourcing"[35], was coined in 2006 by Jeff Howe. He argued that, due to technological advances, the gap between professionals and amateurs has diminished and businesses ought to take advantage of the collective talent and intelligence of the public, either within or outside the organisation, by crowdsourcing operations rather than outsourcing them [259].

What Howe proposed was a way for businesses to copy or learn from the mass collaboration model of Web 2.0 and OSS success stories such as Wikipedia and Linux [531] – typically by leveraging Web technologies. This original interpretation of crowdsourcing and the underlying business model have received a lot of criticism. For example, in a 2007 interview Wikipedia founder Jimmy Wales said [341]:

> « *I find the term 'crowdsourcing' incredibly irritating.* [...] *Any company that thinks it's going to build a site by outsourcing all the work to its users not only disrespects the users but completely misunderstands what it should be doing. Your job is to provide a structure for your users to collaborate, and that takes a lot of work.* »

Indeed, many fundamental questions can be raised. Are crowds at all capable or skilled to carry out the work of professionals? What about the quality of the work delivered? Who takes responsibility if things go wrong? The economics of the model can also be questioned. Is setting up a crowdsourcing system – requiring both a technological infrastructure and participants – managing the crowd – who may be spread across the world, speak different languages and live by different time zones – and dealing with possibly lower-quality results, really cheaper than just outsourcing the work to a subcontractor? Finally, ethical questions can be raised. Will professionals not feel threatened in their occupation? Does it not reek of exploitation when tasks otherwise carried out by (well-)paid personnel are left to a faceless crowd that works for much less or even for free[36]?

[34] For example, humans still outperform computers at tasks such as image recognition and classification.

[35] The practice in which businesses contract work to external parties, possibly in remote, low-cost places.

[36] And likely without written contracts or legal protection.

Despite such controversies, the term has caught on and is now widely used to refer to a broad range of both commercial and non-commercial initiatives, many of which are highly successful. In a recent article Doan et al. present a survey of usages of crowdsourcing on the Web [133]. They take an open perspective – independent of underlying business models – and define *crowdsourcing systems* as « [systems that] *enlist a crowd of humans to help solve a problem defined by the system owners* ». Pretty much all Web 2.0 and social network sites mentioned above, along with OSS projects, peer-to-peer file sharing systems, product review and recommendation systems on retail sites like *Amazon* and *volunteer computing/thinking* projects (see below), are cited as examples of such systems. Doan et al. classify these crowdsourcing systems according to a number of dimensions:

- Whether the collaboration among contributors is explicit or implicit;
- Whether or not the system is standalone or "piggybacks" on another;
- Whether or not users must be (actively) recruited;
- What the users are actually doing.

Moreover they identify five key challenges which operators of such systems face:

- How to recruit contributors;
- What can they do;
- How to combine their contributions;
- How to manage abuse;
- How to balance openness with quality.

Wikipedia provides plenty of evidence on the last two challenges (as well as the others). In principle, every entry in the encyclopaedia can be edited by anyone, yet this also opens the door to acts of vandalism and biased or otherwise inaccurate contents. Therefore, the website employs a range of review, discussion and locking[37] mechanisms to ensure quality and avoid abuse [590]. Overall, this seems to work well, as shown by a 2005 study published in *Nature*, which found that Wikipedia achieved a level of accuracy close to that of *Encyclopædia Britannica* and had a similar rate of serious errors [208].

It is also interesting to note that software platforms and frameworks have been created to facilitate the development of online crowdsourcing systems. Such a framework is for example provided by Amazon's *Mechanical Turk* [13], an online crowdsourcing marketplace where "requesters" can define *Human Intelligence Tasks* (HITs) and then attract a crowd of contributors (volunteering or paid) to execute them.

[37] For instance, entries on controversial issues are sometimes temporarily locked or more closely monitored.

2.4.3.3 Citizen science

The practice of *citizen science* is a form of crowdsourcing in which efforts of the, or a, crowd contribute to a scientific goal or project. It can be defined as initiatives in which non-professional or amateur scientists actively participate, either individually or in groups, in scientific tasks like observation, measurement, classification, analysis, computation or dissemination [92, 480]. Typically such initiatives are set up and coordinated by professional scientists and the participating non-professionals are usually volunteering citizens (i.e. *citizen scientists*) who may lack a formal education in the relevant discipline and have received only limited ad-hoc training. A noteworthy remark was made by Haklay [237]:

> « [...] *by definition, citizen science can only exist in a world in which science is socially constructed as the preserve of professional scientists in academic institutions and industry, because, otherwise, any person who is involved in a scientific project* [could be] *considered a scientist.* »

The *Christmas Bird Count* is an often mentioned early example of citizen science. This annual campaign, first organised in the USA in 1900, now involves tens of thousands of amateur birdwatchers in over 25 countries [92, 364, 480]. In the UK, the British Trust for Ornithology has been organising similar volunteer-based surveys since 1932 [57, 480].

Like other forms of crowdsourcing, citizen science is not without controversy. The biggest challenge it faces is a clash with the current culture of science, which is characterised by the pursuit of professionalism, precision and accuracy and elimination of uncertainty [237]. Professional scientists may doubt the ability of amateurs to do "real research" – i.e. to do part of "their" work [92]. Consequently they may be reluctant to accept findings of studies involving citizen scientists. Still, the practice has – maybe even before 1900 – and continues to be applied in many fields – including, besides ornithology, meteorology [93] and astronomy [563, 600] – for tasks that require, or benefit from, manpower, resources or spatio-temporal reach beyond what can be realised by scientific personnel and their tools.

The Internet, the Web and other technological advances have brought new possibilities and renewed interest for citizen science [243]. On the one hand, technology has revolutionised existing forms of citizen science. For example, the *eBird* platform leverages Web technology to facilitate and extend volunteer-based ornithological surveys [506]. On the other, technology has enabled new forms of citizen science. For instance, the Internet has made it possible for citizens to contribute to scientific endeavours from the comfort of their own home. *Volunteer computing* and *volunteer thinking*, which we will discuss below, are examples of such indoor, desk-bound forms of citizen science[38].

[38] Such Internet-based forms of citizen science are sometimes referred to as *citizen cyberscience* [86, 395].

2.4.3.3.1 Volunteer computing & thinking

Volunteer computing is a form of distributed or grid computing in which volunteers contribute computing resources, typically of home PCs or gaming consoles, to (mostly) scientific projects requiring large amounts of computing power or storage. One of the earliest and most famous examples is the *SETI@Home* project, which in 1999 embarked on a, so far unsuccessful, search for signs of extra-terrestrial intelligence by crunching observational data collected by a radio telescope [19]. Another well-known example is *climateprediction.net*, which runs climate simulation models on PCs [11, 12, 89, 490][39]. In recent years many other volunteer computing projects have been launched[40], focused on everything from finding large prime numbers [481] to protein folding [477][41].

The basic concept of all these projects is the same: volunteers install a client application on their machine(s). This client regularly downloads work packages (i.e. a chunk of data to be analysed) from a central infrastructure, performs the work whenever the machine has resources to spare and can usually visualise (e.g. as a screensaver) the progress in a way that is somehow interesting or appealing (at least for a while). The completion of a work package is rewarded with credits, without monetary value. Accumulated credits are listed on leader-boards on the project website and usually participants can form teams to pool their credits and compete with other teams. Such elements of competition, reputation, showing off (*"Look how fast my overclocked PC is!"*) and sense of belonging, have been shown to be motivating factors, albeit not for everyone. Other motivations include an interest in the goals of the project or science in general and the gratification felt from making a small but meaningful contribution to it [89, 391, 392].

The related concept of *volunteer thinking* applies to projects that employ the cognitive power of volunteering citizens themselves, instead of, or in addition to, the computing resources of their machines[41]. One such project is *Galaxy Zoo* [432, 433], which lets people classify galaxies and celestial bodies photographed by telescopes. Perhaps the most interesting practice in volunteer thinking is the packaging of cognitive tasks as entertaining and challenging games. An early example of such a *game with a purpose* (GWAP) [559] was the *ESP Game*, which tackles the problem of image recognition and metadata generation by letting pairs of players (who may well be on opposite sides of the globe) independently suggest labels for an image. Only labels suggested by both players are retained and are rewarded with credits [558]. Another example is *Foldit*, which presents the problem of protein folding as an online multiplayer puzzle game that lets players quite literally fold proteins "by hand" [99]. Like in volunteer computing, volunteer thinking and GWAP projects exploit elements of competition and reputation to encourage participants.

[39] Thanks to efforts of my former colleagues Hanappe & Beurivé from Sony CSL Paris, part of the climateprediction.net software can now also run on PlayStation 3 consoles [242].

[40] This is in part thanks to *BOINC*, the distributed computing infrastructure originally developed to underpin SETI@Home, which was opened-up for other projects to build on [18].

[41] Besides providing scientists with computing or cognitive power, volunteer computing/thinking projects also allow to raise public interest for scientific research and awareness of its results [11].

2.4.3.3.2 Geographical citizen science

The term *geographical citizen science*, introduced by Haklay, refers to projects in which the collection of spatial/location data forms an integral part of the activities carried out by citizen scientists [237]. In fact, most projects involving fieldwork fall in this category, because observations – e.g. of birds – or measurements are usually coupled with an identification of the location where they took place. New technologies have revolutionised this type of citizen science in two ways. On the one hand, accurate and easy to use positioning technology has become mainstream in the form of portable GPS receivers, either as standalone devices or integrated in personal navigation devices and smartphones [237]. On the other, the barrier to map making, spatial analysis, interoperability of geographical databases (*mash-ups*) and the use of geographical information systems (GISs) in general, has been lowered thanks to *GeoWeb* applications and services [240]. Examples include *Google Maps/Earth* [215, 216], *KML* [298], *OpenLayers* [398] and *CloudMade* [91].

A particularly interesting practice, at the intersection of geographical citizen science, user-generated content and crowdsourcing, is *volunteered geographic information* (VGI), which is defined as the harnessing of tools to create, assemble, and disseminate voluntarily supplied geographic information [213]. For centuries the gathering of geographical or spatial data has been reserved to official agencies, staffed with geographers and other professionals. But, thanks to the abovementioned technological evolutions, much of it is now within reach of volunteering citizen scientists. The most prominent example of this phenomenon is *OpenStreetMap* (OSM) [239, 399], which is a *collaborative mapping* project that aims to create a free editable digital map of the world. Much like Wikipedia, OSM relies on commons-based peer-production and all data collection, editing and integration is done by volunteers. Recent studies in the UK have shown that OSM's coverage is rapidly growing and that the data is fairly accurate in comparison with datasets created by Ordnance Survey, the UK's official cartographic agency [236, 238].

A related emerging practice within Geography and GIScience is *Participatory GIS* (PGIS). By leveraging user-friendly and integrated applications of geo-spatial technologies, PGIS seeks to empower communities in spatial decision-making processes, such as urban planning, land-use planning or environmental conservation [100, 137].

2.4.3.3.3 Community & street science

From our perspective, the most interesting initiatives are those where members of local communities carry out scientific work to construct an evidence base, raise awareness and establish civic action plans to deal with environmental or social problems they face in their area. This form of citizen science has been called *community science* [237] or *street science* [102]. Here, citizens' primary motivation to participate is a concern about local issues, rather than an interest in science per se. It is even conceivable that the initiative is taken by community members themselves, rather than professional scientists. However, even such bottom-up, grassroots initiatives usually require and seek assistance of professional scientists or facilitators. Hence, community or street science projects are less about crowdsourcing (citizens working *for* scientists) and more about collaboration (citizens working *with* scientists) or service to society (scientists working *for* citizens).

In *Street Science: Community Knowledge and Environmental Health Justice* [102], Corburn argues that when environmental health problems arise in a community, policy- makers can achieve better results if they are able to reconcile and combine residents' first- hand experience and local knowledge with recommendations by professional scientists. Regarding the tension with professionalism, he notes [102: p. 3]:

> « [street science] *does not devalue science, but rather re-values forms of knowledge that professional science has excluded and democratizes the inquiry and decision-making processes.* »

Based on studies of actual street science projects, Corburn concludes [102: p. 201]:

> « [a community's] *political power hinges in part on* [its] *ability to manipulate knowledge and to challenge evidence presented* [by authorities] *in support of particular policies.* »

Hence, although they may be hesitant to accept or act upon findings of community/street scientists blindly, well-meaning policymakers should value and stimulate their efforts.

Maps are powerful tools to expose environmental or social injustices and to influence policymakers in community-affecting decisions [102: pp. 173–199, 142]. Consequently many community/street science projects involve spatial data collection and map making, and can thus also be situated in the scope of geographical citizen science (VGI and PGIS).

A successful example are the *bucket brigades*. In these grassroots air pollution monitoring campaigns, community members sample air using simple plastic buckets for analysis by a laboratory [237, 411]. The campaigns are supported by Global Community Monitor, a California-based organisation which trains communities around the world to help them understand the impact of the fossil fuel industry on their health and environment [526]. Also exemplary are the campaigns set up by The Food Trust, a community action group based in Philadelphia that fights obesity and malnutrition in urban areas. Maps are used to illustrate the relation of obesity-related deaths and lack of access to fresh produce markets [142, 524]. A third example are the noise measuring campaigns conducted by members of two London communities impacted by airport and industrial activities. The involved citizens were assisted by professionals from University College London and London 21, a charity organisation. The campaigns were able to bring the noise pollution problems these communities face to the attention of policymakers [151, 193, 324, 325]. In [102], Corburn studies four street science projects that took place in Brooklyn, NYC.

2.4.3.4 DIY culture

Do-it-yourself (DIY) communities are another source of possibly relevant practices. DIY is an activity of all times, stemming from the need to repair or refurbish existing but incomplete or broken objects or systems. The mass production-consumption economy does not stimulate self-reliance, but still people keep on creating or repairing things by themselves, even though professionals could do it for them. One of the motivations within DIY culture is the rejection or avoidance of mass consumption and the embrace of more sustainable lifestyles. Today there is a renewed interest for DIY culture and practices, among practitioners (i.e. DIYers) as well as (HCI) researchers and designers [70, 131, 306, 418]. Thanks to technological advances, DIY enthusiasts can now easily document and share their projects with likeminded people, and set up collaborations with DIYers from across the world. Examples of such new (online) DIY communities are *Instructables*, *Dorkbot*, *Adafruit* and the *Arduino* community [306]. The "green" motivations of DIY practitioners, and their inclination to take matters into their own hands, makes them an interesting audience to recruit from for participatory environmental monitoring campaigns and possibly also for the creation of tools that can be used in those campaigns, such as the air sampling buckets mentioned above. Online DIY communities may also serve to disseminate instructions on how to (re)create such tools and how to organise such campaigns.

2.4.3.5 Openness in governance

In the last decade authorities at various levels have embraced and adopted new policy and communication instruments enabled by the Internet and the Web (2.0). Authorities have realised that advances in ICT create opportunities to make governance both more efficient and more transparent, inclusive or "open". This trend is commonly referred to as *e-government* or *digital government* [581] and is the subject of an active research field at the intersection of political and computer science. Interesting recent publications on the matter are *The Power of Information* [338], a 2007 report commissioned by UK government; *Web 2.0 in Government: Why and How?* [403], a 2008 report by the European Commission's Joint Research Centre; and *Wiki Government* [389], a 2009 book by Beth Simone Noveck, the former deputy-CTO of the U.S. government.

A central theme in this area is *open data*, which is the idea that various types of data – typically in an academic or governmental context – should be freely available to everyone to (re)use and republish as they see fit, without copyright, patents or other restrictions. In this spirit several authorities have, or plan to, set up portal websites that centralise datasets on diverse aspects of policy and governance, for the benefit of scientists and citizens alike [207, 535, 546]. Noveck argues that such websites are a powerful tool to enhance the transparency and accountability of democratic governance [303, 389]. For our purposes a particularly interesting initiative is *Eye on Earth*. This is an open online platform, launched by the European Environment Agency, that leverages GIS and cloud computing technology to enable citizens to explore, analyse, visualise and reuse environmental data made available by various authorities in and outside Europe [169].

These attitudes and technologies can also enable scenarios in which governments accept and even request data gathered and shared *by* citizen(scientist)s, instead of the other way around. Regarding the potential of citizen-collected data, Noveck wrote [389: p. 21]:

> « [...] *the "single point of failure" in government can be transformed through new mechanisms for obtaining expertise. Decision-making is currently organized around the notion that the government official knows best. In reality, agencies make decisions every day without access to the best information or the time to make sense of the information they have. Citizen participation traditionally focuses on deliberation but, in the Internet age, it will not be as successful as collaboration in remedying the information deficit. The broader mandate is to use technology to upend the outdated theory of institutional expertise and replace it with collaborative practices for gathering and evaluating information and transforming raw data into useful knowledge.* »

Today, authorities seem increasingly willing to embrace such possibilities and operational examples are appearing. For instance, Eye On Earth allows users to post personal ratings of environmental conditions in their local area [169]. Other examples are *FixMyStreet* and *BuitenBeter*, which apply Web and mobile sensing technology to let citizens report problems (e.g. potholes, litter, graffity, broken street lighting, etc.) to their city or community council [302, 361, 610]. While both platforms were not created or commissioned by authorities, many local councils have adopted them as a channel for citizen input.

2.4.4 Summary

In this section we argued that the three principal ingredients for a low-cost, participatory approach to large-scale environmental monitoring and governance are participants, affordable sensor technology and collaboration practices. Based on recent surveys we showed that there is an high level of concern in the (European) public opinion about issues such as climate change and the environment in general. This makes us optimistic that participants for the systems we have in mind can indeed be found and mobilised. Next, we explained that the growing popularity of smartphones, mobile broadband and mobile applications has led to the emergence of a low-cost, ubiquitous sensor platform. Finally we gave an overview of new practices for human interaction and expression, (content) creation, gathering and dissemination of (scientific) data or knowledge, and public participation in governance. These elements represent a unique opportunity to enable participatory environmental monitoring and governance on previously unattainable scales.

2.5 Our vision

The core of our vision for environmental monitoring and governance, as put forth by Steels [493–495], is formed by *community memories*, a concept dating back to the 1970s.

The *Community Memory* was the name of the first public computerised bulletin board system, established in 1973 in Berkeley, California by Lipkin et al. [94, 521]. The system was conceived as an information and resource sharing network aimed at strengthening the Berkeley community by linking a variety of local organisations. Soon it was generalised to a kind of information flea market, providing unmediated, two-way access to message databases through public computer terminals. Users demonstrated that it was a general communications medium that could be used for art, literature, journalism, commerce, and social chatter [464, 521, 577]. In retrospect, the Community Memory can be seen as a forerunner of today's online communities and social networks [42, 482]. A product of the 1960-70s' countercultural movement, the system was inescapably political in its aim to empower local communities and support social change [199: pp. 42–43, 454].

After having introduced the community memory concept in the context of open expert systems in the mid-1980s [491][42], Steels, inspired by Ostrom [404, 409] and others, reinterpreted it in the scope of sustainable commons management in 2007 [493–495]:

> « *A community memory is a medium for recording and archiving information relevant to a commons that is managed by a community and for diffusing this information among members or communicating it to those threatening the commons and thus the community. All members making up the community typically have access and are allowed to upload, download or inspect information. Once the information is there it becomes possible to 'add intelligence' to the system in various ways, for example by creating maps* [...], *by explicating dependencies between information items in order to bring out trends and predict future evolutions, by simulating future states of the world, etc.* »

In this interpretation, a community memory supports a community in building an evolving representation of itself and the commons it is concerned with. This representation serves as an evidence base for discussion and interaction within the community and with external stakeholders, with the goal of sustainably governing the commons. Today, the opportunities identified in the previous section make it possible to put this into practice. Figure 2.1 illustrates the position such a community memory, enabled by contemporary ICT and associated practices, could take in a society where a citizen community, (local) authorities, scientists and other parties have a stake in the management of a commons.

[42] In that context, Steels defined a community memory as « *a knowledge-based system supporting the communication of an evolving body of knowledge among experts* » [491].

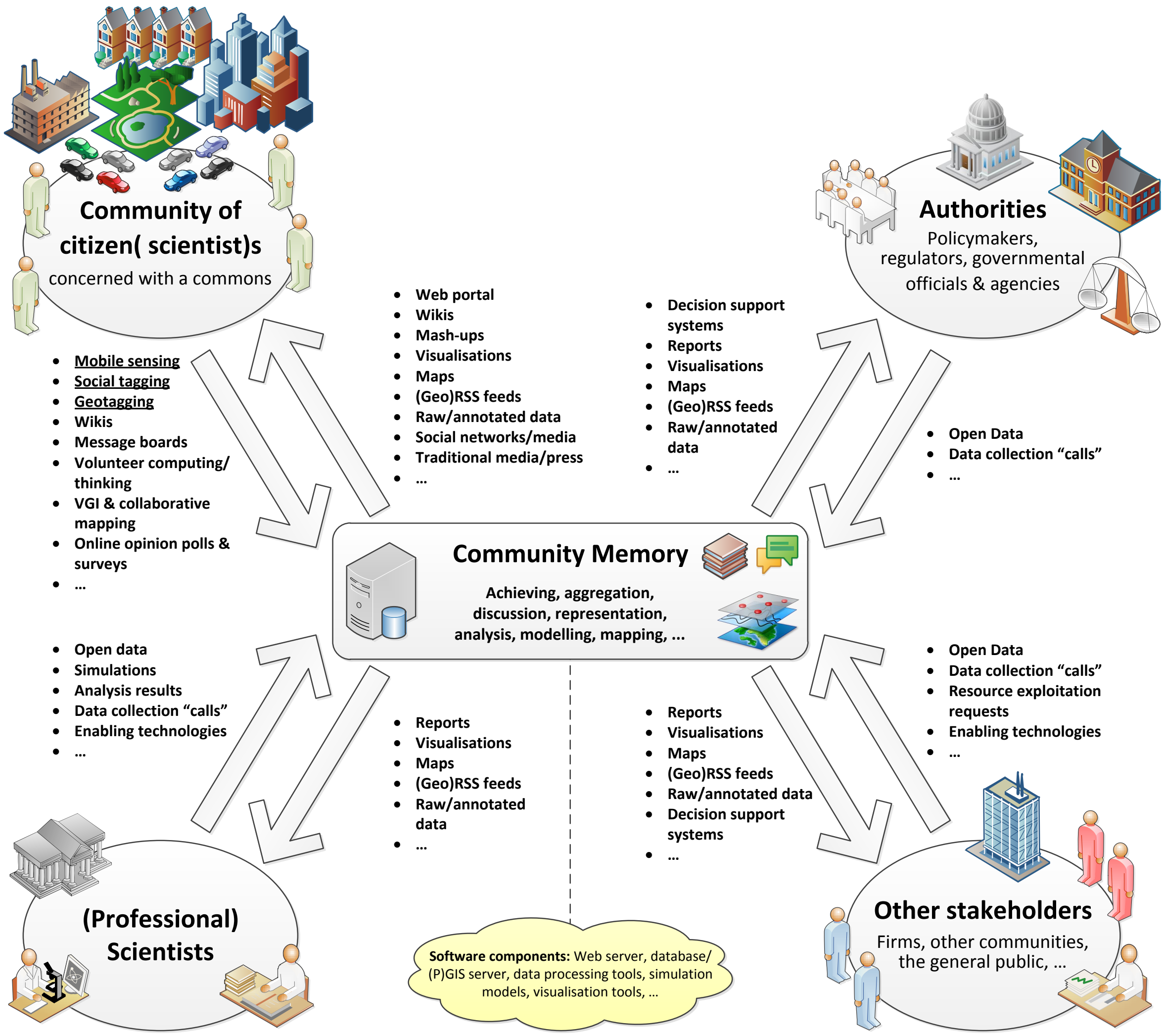


Figure 2.1: Flows of information, queries and resources in and out of a community memory

2.5.1 Community memory scenarios

The scenarios captured by figure 2.1 represent an extended and updated take on the vision presented in [493–495]. To clarify these, we discuss the role of the different stakeholders, relevant technologies or practices, and supported models for commons management:

Community & Commons

Community memories (CMs) are intended for real communities of real individuals, not diffuse groups that flock anonymously through the Internet and have no real state in the management of a commons. Consequently there must first and foremost be a community and a commons to be managed. The community is usually formed by fellow-citizens of a neighbourhood, village, town or city. There may be some existing organisational structure, for instance an association of likeminded neighbours or another type of (local) NGO. On the other hand, it is also conceivable that a community takes shape as part of the creation and use of a CM, or that an existing but loose movement is reinforced by it. Typically the community is relatively small and the duration of a CM project is limited in time – i.e. long enough to resolve conflicts straining the use of the commons [493–495].

In order to function as a community, it is necessary that members recognise each other as individuals and that they meet face-to-face. Such meetings help to create the kind of trust and common ground that is required to self-organise the group's activities (e.g. data collection campaigns). It is also important that identity cannot be hidden and all actions can be traced back to those who carried them out. In fact, a CM is the opposite of a so-called *smart mob*, defined by Rheingold as « *people who are able to act in concert even if they don't know each other* » [450]. With a CM, people are assumed to act in concert because they share a common goal – i.e. the management of a commons in a fair and sustainable way [493–495].

The commons in question can be any CPR, as per the definition used by economists. Besides the examples given in section 2.3.1.2, a more liberal reading could include resources like sanitation, green space in urban areas, space on the road or on public transport, accessibility of public spaces (e.g. for the disabled), the *soundscape* of a neighbourhood (see chapter 4), bandwidth on data networks, intangible assets such as a community's reputation or political clout, and even their general *quality of life*, involving various commons in a specific socio-economic and cultural context.

The CM, and more specifically the information it archives, can itself be seen as a commons: it is a resource, shared by the members of the community, of which (ab)use or (mis)appropriation by some can reduce availability or utility to others. Therefore, this *data commons* [106] should be legally and physically accessible to all members and the community should establish institutions to regulate proper use.

Data collection and management by and for the community

Community members use diverse ICT tools and practices to collect various data and information relevant to the management of the commons. Of particular interest is **environmental monitoring** by means of mobile sensing and social tagging.

Mobile sensing lets citizens monitor various aspects of the(ir daily) environment – as well as the behaviour of community members and other stakeholders within that environment – in a fully distributed fashion using relatively cheap, off-the-shelf mobile phones, possibly complemented with external sensors. This can involve collecting quantitative measurements of physical parameters (e.g. concentration of pollutants in ground water samples), qualitative descriptions (e.g. the water's smell, taste or visual appearance) or media fragments such as photos and audio or video recordings. If location is relevant, the data can be automatically **geotagged** using GPS. Mobile broadband (or alternatively SMS/MMS) allows the material to be uploaded, possibly in real-time, to a central repository – i.e. the CM.

Social tagging is a practical, low-barrier way to collect qualitative (meta)data that supports the organisation, exploration, analysis, interpretation, and understanding of quantitative data or media fragments. Tagging takes little time – making it suitable for mobile usage – and, thanks to the Web, many people are already familiar with it. In a mobile sensing application for in-situ environmental monitoring, it can be used to identify (suspected) pollution sources, clarify spatio-temporal or human context, attract attention to abnormal situations, convey personal opinion, etc. Once the material is uploaded, (other) community members can supply additional tags via a Web interface. While the motivation to tag resources in Web 2.0 services like Flickr already go beyond purely organisational purposes (see section 2.4.3.1.2), social tagging serves an even broader purpose in a CM context. Here, tags are not only aids for future navigation, but also an intrinsic component of the representation-building process [493, 494][43]. Augmenting "raw" material with semantic, contextual and subjective information, allows it to be archived in a more organised fashion *and* disseminated more effectively with respect to the community's goals, for instance by generating maps in which both physical (e.g. measured pollution levels) and subjective parameters (e.g. perceived annoyance) are represented.

Local, community knowledge can also be collected by leveraging other Web (2.0) technologies and crowdsourcing practices, such as wikis, message boards, online opinion polls and surveys, VGI/collaborative mapping using GeoWeb services, etc.

Once data is collected and uploaded it is archived by/in the CM. The CM should enable the community to collaboratively manage the archived material over time. For instance, a Web portal could offer tools to review, edit, discard (e.g. when considered inaccurate) and structure content – similarly to Wikipedia or OSM. Collaborative data management is an integral part of the representation building process and differences of opinion or conflicts of interest within the community may come to light here. Therefore the CM should facilitate discussion, (counter) argumentation and joint reviewing, both online and in the course of meetings.

Aggregation, analysis & modelling

Tools for modelling the environment and predicting the effects of change are critical ingredients to support decision making among opposing stakeholders in commons management [549]. Therefore, a CM should facilitate, or (partially) automate, data aggregation, analysis and modelling, such that material from different sources can be combined and new inferences or predictions (using simulations) can be made. It is conceivable that CM systems distribute part of this computationally or cognitively intensive work to the community via volunteer computing or thinking [493].

[43] Freely chosen tags may also be complemented by selecting *descriptors* from a pre-fixed vocabulary [494].

Dissemination of information

The CM should help the community to disseminate information through multiple channels and in diverse forms, such that the community's message reaches, and is understandable to, all concerned parties – i.e. the community itself, the general public, authorities, scientists, firms, or any other stakeholders. This can involve (semi-)automatically or manually generated maps, other visualisations, RSS feeds, reports, press releases, or posts on blogs and social networks. Additionally, in order to maintain a level of transparency and legitimacy, a CM should allow different stakeholders to access both raw and annotated (tagged/edited/structured) data.

Using social networks as a dissemination channel may help to attract new members via existing social connections. Newcomers can bring different opinions and new data to the group, which may either support or counter the case that is being made.

Governing the commons

In section 2.3 we first discussed the "tragedy of the commons" and Ostrom's proposed solution for collective CPR governance *by* appropriators. Next, we saw that authorities intend to tackle environmental issues, including CPR problems, in participation *with* citizens by adopting an open governance model (cf. section 2.4.3.5) and by raising awareness. We believe CMs can play a central role in both cases.

There where a commons is collectively governed – along Ostrom's design principles (see box 2.1) – we envision CMs to support the establishment of participatory, low-cost monitoring (e.g. by means of mobile sensing) and possibly sanctioning schemes, and the reaching of collective-choice arrangements through open discussion. Note that appropriators can include both the citizen community itself (or a subset of its members) as well as other stakeholders (e.g. firms) exploiting the same commons.

There where a commons is governed by a democratic authority with the participation of a citizen/appropriator community – and on the condition of recognition, by the authority, of the CM and (the rights of) the community itself – we envision CMs to serve as a platform for public participation in all aspects of CPR management and general environmental policy (monitoring, decision-making, internalisation of costs, sanctioning, etc.), and as a multidirectional communication medium to raise and maintain awareness of environmental issues among all stakeholders. Authorities can use this platform to share data with the community, in the spirit of open data, and to consult or request citizen-collected data. In cases where there are no conventional assessment efforts undertaken by authorities or official agencies[44], environmental monitoring carried out by citizens, through mobile sensing and social tagging, can serve as a low-cost – yet not free – alternative. But when there *are* existing efforts, it should be seen as complementary rather than a replacement.

[44] Either because there is no budget or no expertise, or because it is not seen as a priority.

For instance, to assess pollution levels[45] over space and time, fieldwork[46] and modelling by officials remain relevant and necessary. However, tapping the potential of the community of citizen(scientist)s, can lead to massively more quantitative and qualitative data, with denser and broader spatio-temporal coverage[47]. Even though they may have a lower accuracy or credibility[48], large amounts of actual measurements of personal exposure to pollution[49], could be a welcome addition to accurate but sparse pollutant concentration measurements made by officials and/-or simulated levels obtained from models. Hence, we are convinced that combining conventional and participatory assessment methods can lead to better environmental policies that take the opinions and needs of local communities into account[50]. Because citizens have intimate knowledge of patterns in their local area, they are likely to spot anomalies well before officials do. With mobile sensing and a CM they can independently or collectively assess perceived problems, without having to wait for officials. Based on community reports, authorities can then decide to start official, more accurate, inquiries. Moreover, citizen-collected data could be used as an additional input for official simulation models. Additionally, citizens' participation in the assessment process and direct access to its results[51], will potentially increase their awareness of environmental issues, which may in turn stimulate the adoption of more sustainable behaviour and their acceptance of unpopular, yet necessary, measures taken by authorities – especially if they "feel heard" as well.

The role of scientists

Community memories enable new kinds of collaborations between citizens and scientists. As noted by other authors [71, 106, 416], mobile sensing technology in itself already constitutes a major new opportunity for citizen science, because it allows (untrained) volunteers to contribute in data collection campaigns, both to study human behaviour and human environments. However, by organising such campaigns around the intermediary platform of a CM, the role, responsibility and benefits of involved citizens can be extended, resulting in community-empowering collaborations akin to the community/street science projects discussed in section 2.4.3.3.3.

We see three, not mutually exclusive, reasons for professional scientists to participate in a CM scenario. The first reason is in line with conventional citizen science: scientists who want to study an environmental or social phenomenon may enlist citizens to help them. Here, it is useful to look for an existing community that has an interest in the topic of the study, and, in the future, may already use a CM-like

45 And associated health risks.

46 Using professional/dedicated measuring equipment, sensor networks, questionnaire surveys, etc.

47 Possibly including indoor or private places, that are often ignored in modelling or inaccessible to officials.

48 Due to the use of cheaper equipment and possibly a certain bias or a lack of skill among citizens.

49 Enabled by the people-centric measuring perspective.

50 This argument is also supported by the work of Corburn [101, 102] (see section 2.4.3.3.3).

51 To both individual measurements made in the field (in real-time) and aggregated datasets in the CM.

system. Scientists can then request access to previously collected data, propose a new data collection campaign, and may help the community to set up a CM for the benefit of both parties. The second reason applies to scientists who accept an invitation of an existing community to help it reach its goals, for instance by providing methodological support in data collection, analysis and representation. This way the community can increase the credibility and legitimacy of their arguments, and the scientists can get access, via the CM, to large amounts of data and results which can be used for further research and publications. The third reason is found among scientists with an interest in technology-mediated human collaboration and interaction, either from a social science or an engineering perspective[52]. By helping actual communities to deploy CM systems, such researchers can study the potential and usage of CMs for/by specific groups facing specific problems, as well as the requirements for supporting technologies and the nature of associated practices.

Bringing CMs into practice

In concrete CM-enabled collaborations between citizens on the one hand, and authorities or scientists on the other, different choices can be made regarding the initiative, ownership and (de)centralisation. While not explicitly referring to the CM concept, Cuff et al. provide interesting insights in these matters [106]. In principle the initiative to start a CM/mobile sensing project, and deploy the necessary ICT infrastructure, can come from the community itself (i.e. *bottom-up*), but this probably requires user-friendly, reusable software to be available, possibly provided by scientists. Alternatively, authorities could take the initiative to help specific communities by setting up CMs for them (i.e. *top-down*). Yet when the initiative – and/or the ownership/funding of the used ICT infrastructure (e.g. servers and network bandwidth) – does not lie with the community itself, ownership and control over collected information (i.e. the data commons), may have to be shared as well. Nevertheless, in working with citizens to carry out data collection tasks, officials and academics should be prepared to move away from the traditional centralised model, in which they have full control over where, when and which data is collected and how it is processed, represented and diffused.

Community memory projects will probably always be non-profit ventures [495]. Nevertheless, it is conceivable that commercial enterprises would sell software or services that can be used to set up a CM system, or coach communities to use one. However, the communities most likely to benefit from a CM[53] typically lack the finances to buy such products or services – unless they are somehow sponsored (e.g. by a governmental body or an NGO). Hence, there is a need for free, preferably open source, CM software and voluntary coaching by scientists or facilitators.

[52] For instance in the field of HCI or CSCW.

[53] See section 2.5.2 for some examples.

2.5.2 Examples & implications

In [493–495], Steels discusses a number of examples of deployed community memories. One case study is about the work of anthropologist Jerome Lewis, who set up a CM for the Mbendjele pygmies living in the rainforest of the Congo basin. Lewis' system – based on PDAs, GPS receivers and (P)GIS technology – is designed to empower this illiterate[54] community in their struggle to protect forest areas which are important to them from destruction by logging companies [257, 318]. The other case studies are about CMs set up by Eugenio Tisselli, co-author of [494, 495], for taxi drivers in Mexico City, disabled people in Barcelona and motorcycle couriers in São Paulo. These projects were part of Tisselli's *zexe.net* (now *megafone.net*) initiative [528], which since 2003 invites communities on the fringe of society in cities in Latin-America, Europe and North-Africa, to express their experiences and opinions via face-to-face meetings and mobile sensing. A mobile app lets participants make audio recordings and photos, tag them, and directly publish them on a CM website. The goal is to amplify the voice of individuals and groups who are overlooked or misrepresented in mainstream media and struggle to manage tangible (e.g. the right to legally operate a taxi, or to access public spaces in a wheelchair), or intangible (e.g. their reputation in the eye of the general public) commons [494, 495].

From these case studies, Steels and Tisselli distil some initial conclusions regarding the organisation of CM projects and the design of CM systems. In line with Ostrom's views, they argue that technology in itself accounts for only a relatively small part of the success of a CM project[55]. Instead, setting up the social organisation around the CM appears to be the biggest factor in the success of a project. This task requires excellent organisational and communicational skills and insights in the community's structure and interests. Therefore it is best carried out by social workers with strong ties to the community – or maybe by leader figures who emerge from the group itself and/or are accountable to it. Next, they observe that, because CM projects often deal with people who are not literate in the use of technology, or are even illiterate in the original sense, a fundamental requirement for CM systems is simplicity and ease of use, which implies a design approach that is tailored to the needs and habits of the community in question. Moreover, it is useful to recognise that the level of skill and enthusiasm may vary among community members. For instance, it may be acceptable to expose more complexity in the (Web) interfaces for data management and editing than in those for (mobile) data collection, since the former will likely be used especially by the more experienced or eager participants. Finally, they stress the importance of having two interfaces. The one used within the community should be only accessible and configurable by community members, such that they feel safe to add (personal) information. The system should thus offer a separate interface, and produce separate dissemination artefacts, for interaction with outsiders [495].

[54] Because of the users' illiteracy, the system employs a decision tree with graphical icons instead of tags.

[55] Calling a CM project successful implies that the source of tension around which the CM took shape (e.g. the conflict over the exploitation of a patch of rainforest) is managed, if not resolved [495].

2.5.3 Related work

Our vision for community memories shares elements with ideas and concrete projects or systems presented by others. Here is an overview of the most interesting cases:

★ The community/street science initiatives discussed in section 2.4.3.3.3 can be seen as implicit CM building projects, in terms of intentions, purposes and some applied practices, but not, or to a lesser extent, in terms of applied technologies.

★ While especially focused on collaboration practices, rather than on the role of ICT, Corburn's street science framework [102], which stresses the importance of local community knowledge to improve scientific inquiry and decision-making in environmental health policy, provides methodological and practical support for our vision.

★ In a position paper titled *Participatory Sensing* [71], Burke et al. were the first – as far as we know – to link citizen science with mobile sensing technology. They envisioned "grassroots sensing" campaigns for the purpose of environmental monitoring, public health and urban planning[56], expression of cultural identity and creativity, and natural resource management. We will further discuss this paper in chapter 3.

★ In the already mentioned article *Urban Sensing: Out of the Woods* [106], Cuff et al. examine how emerging sensing technologies may change the way citizens, authorities and scientists collect, manage, share and apply sensor data in urban areas. Moreover, they introduce the aforementioned concept of a *data commons*, which in many ways is similar to our interpretation of a community memory.

★ In a book chapter titled *Citizen Science: Enabling Participatory Urbanism* [416], Paulos et al. explore how the shift in the usage of mobile phones – from communication tool to « *networked mobile personal measurement instrument* » – enables a new form of citizen science which they call *participatory urbanism*. They envision this will help citizens « *to become active participants and stakeholders as they publicly collect, share, and remix measurements of* [what] *matter*[s] *most to them* ». Moreover, they report on pioneering experiments to apply this idea to air pollution monitoring (see chapter 3). In a later paper [417], Paulos sheds light on design strategies for citizen science tools and ethical aspects of the practice. He argues that, to facilitate social change, designers should encourage doubt and debate, rather than promote blind acceptance of fact. In recent work, Paulos and his team study mobile sensing, *persuasive technology*, citizen science and DIY culture from an HCI perspective by experimenting with *sensor probes* in the field [305, 307, 418].

- In [179], Ferron and Massa study the process of collective memory building on Wikipedia in the context of the 2011 popular uprisings in the Arab world. This work seems to indicate that Wikipedia – and possibly other (wiki-based) peer production platforms/communities – has CM characteristics or can be adopted as such.

[56] And the restoration of the link between both [101].

- In [329], Luther et al. present *Pathfinder*, an online collaboration environment for citizen scientists – that sadly is neither freely nor commercially available in any form. While not explicitly referred to as a community memory system, the described software contains many elements we envision for CMs. For instance, it supports sharing, annotating, aggregating and visualising data time-series (called *tracks*), and collaborative data analysis. By means of a user study the authors established that citizen scientists preferred Pathfinder to a standard wiki and that it enabled them to go beyond data collection and engage in deeper discussion and analysis [329].

- FixMyStreet and BuitenBeter (see section 2.4.3.5) can also be seen as CM systems.

- In [1], Aanensen et al. present *EpiCollect*, a platform that combines smartphone and Web applications to facilitate epidemiological, ecological and community data collection. While the proposed system is not a CM, there are similarities in the way mobile and Web technology is applied to support data collection in the field by both professionals and citizen scientists.

The items marked with a "★", in the list above, were direct or indirect sources of inspiration for our vision and the work discussed in this dissertation, the others were not. Additional related work, regarding mobile sensing, is discussed in chapter 3.

2.6 Conclusion

In this chapter we first sketched the societal context in which our work fits, namely the grand effort to find answers to the pressing challenge of achieving sustainable development on global and local scales. Next, we learned that, whether environmental issues, and specifically the need to sustainably manage commons, are tackled by citizen-led or authority-led initiatives, there is a need for flexible, low-cost approaches to environmental monitoring and governance, based on broad public participation and awareness raising. Then we argued that the three principal ingredients for a novel, participatory, ICT-supported approach to this problem are motivated participants, affordable sensor technology and collaboration practices. Recent surveys of environmental concerns in public opinion make us optimistic that participants can be found. Ubiquitous computing technology, notably in the form of smartphones, mobile broadband and mobile apps, creates a new potential for dense, people-centric sensor networks, virtually anywhere people flock together. The last decade has seen the emergence of participatory practices for content creation and annotation, collaborative work, gathering and dissemination of (scientific) data, and open governance. Together, these ingredients represent a unique opportunity for participatory environmental monitoring and governance on previously unattainable scales.

The main contribution of this chapter is the vision we developed around the concept of *community memories*. We see a community memory as an ICT resource that empowers a community by enabling it to archive, discuss, augment, visualise and share information that is relevant to the management of a commons it is concerned with. While possible scenarios and concrete examples are highly diverse, the community's goal is always the sustainable management of a commons, usually in collaboration with, or in opposition to, other stakeholders such as firms, authorities and scientists. To monitor their environment and the behaviour of those that threaten it, at low cost, citizens can use mobile sensing and social tagging. Once data is collected, tagged and uploaded, the community memory system facilitates its management, aggregation, analysis, visualisation and dissemination. Community memories can play a central role in the sustainable exploitation of collectively-managed commons, as well as in the establishment of successful environmental policies – including the management of centralised commons – by authorities in participation with citizens. Additionally, community memories create new opportunities for citizen science.

In view of the sustainability challenge humanity faces, what we propose can be seen as a « *Think global, act local* »[57] kind of approach: by helping local communities to achieve ambitious, yet attainable, goals, we intend to contribute to solutions for global problems.

Environmental monitoring by means of mobile sensing is a cornerstone of our vision. Mobile sensing is the subject of active research in computer science, that has applications well beyond the domain of environmental monitoring or community memories. In the next chapter we take a more detailed look at this field, in order to situate our research and its main artefact, the NoiseTube system, within this growing body of related work, to highlight the sources of inspiration that have helped us to refine our vision into a concrete approach, and to contrast this approach with the choices of others.

[57] This phrase, attributed to Scottish biologist, town planner and social activist Patrick Geddes, became a popular slogan of environmentalism in the late 1960s and early 1970s [594].

Chapter 3

Mobile sensing

3.1 Introduction

From the perspective of computer science, much of the work presented in this dissertation can be situated in the field of *mobile sensing*, which has gained a lot of interest lately. Mobile sensing leverages the technological platform of (smart)phones, mobile broadband and app(store)s. In recent years, many researchers, as well as commercial enterprises, have proposed or developed mobile sensing systems for a wide range of application domains, including environmental monitoring and citizen science.

The purpose of this chapter twofold. On the one hand we provide a critical overview of this field, including its origins, state of the art, and open challenges. On the other, we refine the vision presented in the previous chapter into the concrete approach which we have followed in our own research. This description includes: a detailed set of specifications for mobile sensing systems aimed at participatory environmental monitoring in a community memory context, and a motivated selection of challenges we have chosen to work on in this context.

In this chapter, we first sketch the origins of mobile sensing and its relation with *wireless sensor networks* in section 3.2. In section 3.3 we present the state of the art of mobile sensing research and applications, including a comparison of the main schools of thought, illustrated with examples of related work. Next, section 3.4 provides an overview of the main challenges the field is currently facing. Then, in section 3.5, we concretise our approach and position our work with respect to the design space for mobile sensing applications and open research challenges. Finally, section 3.6 wraps up the chapter.

3.2 Origins

Mobile sensing is rooted in the wider movement of *ubiquitous computing*. Ubiquitous computing is a vision proclaimed by Mark Weiser in the early 1990s [566–568]. In this vision, PCs are no longer the central point of human-computer interaction. Instead, computers, or rather computing itself, would be integrated in everyday objects to allow *anyone* to interact and interconnect with computing services and information resources *anywhere* and *anytime*. Weiser foresaw that this shift would cause an increase of the availability and a decrease of the visibility of computing, because, as he observed [566]:

> « *The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.* »

Today, thanks to the emergence of personal, mobile devices and mobile Internet connectivity, Weiser's vision – especially the availability side – has largely become a reality.

The ubiquitous computing paradigm is also known as *pervasive computing* and *ambient intelligence*, where each term emphasizes slightly different aspects. A closely related vision is that of the *Internet of Things* (IoT), introduced by the International Telecommunication Union in 2005, which adds a fourth dimension to ubiquitous computing: in addition to *anywhere*, *anytime* connectivity for *anyone*, IoT foresees connectivity for *anything* [274]. In an IoT world, which is also increasingly becoming a reality, Internet connectivity will be integrated in more and more objects, buildings, vehicles, etc. In 2009, the European Commission adopted an action plan to help realise the IoT vision [161].

Sensing plays an important part in these visions. We are on a path where various types of sensors are being integrated into more and more everyday devices and objects, enabling software to make (real-time) inferences about more and more aspects of our needs, behaviour and surroundings, and about the world in general. On the individual level, this knowledge is expected to be leveraged by "smart" services that aim to simplify our lives. On the collective level, this knowledge is expected (or hoped) to enable us to increase the efficiency of the services and systems that underpin the modern economy and society – e.g. (public) transport, energy provision, logistics and supply chains, healthcare, etc. – and help us to understand and protect our environment better [161, 274, 566, 601].

3.2.1 Wireless sensor networks

Pioneering research into sensing systems, in the scope of ubiquitous/pervasive computing and the IoT, has been conducted in the area of *wireless sensor networks* (WSNs) – also known as *embedded networked sensing* [106]. A WSN consists of spatially distributed, typically highly-specialised, autonomous *sensor nodes* (also called *motes*) that monitor various physical parameters. Each node is equipped with one or more sensors, possibly some actuator(s), a wireless transceiver, an embedded microcontroller, a power source (batteries or capacitors), and possibly external (flash) memory. Nodes are usually installed in stationary positions. Alternatively, "nomadic WSNs" can be formed by installing nodes on vehicles or by regularly moving them to new places manually [54, 110, 453, 484].

One of the main drivers of WSN research is the goal of increasing the autonomy and spatial reach of WSNs, requiring innovations in hardware, software and protocols. Optimisations of hard- and software aim to make individual nodes as power-efficient as possible, and advances in energy *harvesting*[1] and storage technology further increase autonomy. Moreover, topology-aware, multi-hop communication protocols have been developed to route instructions and data across WSNs in a power-efficient and fault-tolerant manner.

In the past decade, WSN research has moved from confined laboratories to relatively small real-world deployments, often in natural habitats like forests or lakes [106, 107]. Moreover, there has been a focus on WSN deployments in industrial settings [130, 234]. Although reusable hard- and software platforms are available (e.g. [479, 527]), WSN nodes are anything but commodity devices. Building, programming and deploying WSNs for a particular application or context usually requires specialised expertise and is thus only within reach of academics and other professionals (e.g. governmental agencies)[2].

3.2.2 New forms of sensing

In recent years WSN research, and ubiquitous computing in general, has seen a shift towards applications in urban settings [106, 107, 189, 300, 478]. In parallel, the rise of sensor-laden smartphones[3], mobile broadband and, more recently, apps and app stores, has created a new, people-centric and mobile platform for sensing systems [74, 106]. These evolutions have led to new forms of sensing research and applications.

The research field and the types of applications that have emerged have been referred to with many terms, such as: *people-centric sensing* [74, 75], *participatory sensing* [71], *opportunistic sensing* [74], *urban sensing* [106], *mobile sensing* [258, 294], *community sensing* [304, 310], *citizen sensing* [71, 417], and *mobile phone sensing* [310][4]. These refer to largely overlapping concepts, but each stresses other aspects or choices within a shared design space. For convenience, we refer to this ensemble simply as *mobile sensing*.

[1] Also known as *energy scavenging*, this is the process of deriving energy from local, ambient sources such as solar energy, thermal energy, wind energy, vibrations or mechanical strain and RF radiation [505].

[2] We should note that there have been attempts at designing end-user programmable WSN nodes [357].

[3] Refer to section E.1 for a discussion of what exactly constitutes a smartphone.

[4] Or combinations thereof.

In mobile sensing systems, smartphones – or cheaper, yet usually Internet-connected, mobile phones – fulfil a role that is similar to that of sensors nodes in WSNs. When phones do not have the required sensors for a particular sensing application – e.g. measuring gas concentrations – they are complemented by external sensor units that are either physically attached (e.g. via USB) or wirelessly connected (e.g. via Bluetooth) [88, 316].

In most sensing scenarios it is necessary to know where and when sensor data is collected. In stationary WSNs the geographic position of each node must only be determined once. Because people and their phones tend to move about, mobile sensing systems require a mechanism that regularly establishes their current position. Nowadays, smartphones come with built-in GPS receivers that provide accurate location (and time) estimates. However, before GPS receivers became so common, and because GPS only works outdoors and consumes a lot of power, alternative and complementary mobile positioning technologies have been developed, mostly based on triangulation between cellular network antennae and/or Wi-Fi access points [53, 114, 252, 448]. In today's GPS-equipped smartphones such systems are still used[5] to provide indoor positioning and to augment or stand in for GPS in situations where it does not perform optimally (e.g. in *urban canyons* [252]), consumes too much power or when accuracy requirements are less stringent.

Some early mobile sensing systems were similar to conventional WSNs. For instance, [21, 22, 254] discuss experiments in which smartphones served only as GPS receivers and communication relays and were packaged together with Bluetooth-connected external sensors, to be installed on vehicles and street furniture. Compared to conventional WSNs, such "hybrid" setups may be cheaper and less complicated to deploy since they are partially based on off-the-shelf hardware and leverage the existing cellular network infrastructure. But apart from that, the phones add little extra, mainly because no mobile phone *users* are involved. As discussed in section 2.4.2, the insight that smartphones can be used to study human behaviour *and* human environments from a *people-centric* perspective (and that users can play an active role therein), created interest for sensing systems in which phones, and any external sensors, are carried around by people, usually volunteering citizens, instead of being mounted on buildings, street furniture, trees or vehicles[6].

In table 3.1 we list the most important differences between WSNs and (people-centric) mobile sensing systems. Most differences stem from the hard- and software platforms which are used. The specialised and hard-to-program platforms used in WSNs make them more expensive to deploy on a large scale (and with dense granularity), than mobile sensing systems, which rely on easier to program, off-the-shelf devices, which many potential users already own. However, WSNs have their advantages too. Because WSN nodes are highly optimised, single-purpose devices they have a longer autonomy than smartphones, which, besides sensing, are also used for many other things – such as the primary func-

[5] In most smartphone operating systems, such as iOS [30], Android [226] and Symbian [519], the choice of positioning technologies is made largely transparent to application programmers.

[6] A hybrid approach in which phones are mobile but external sensors are stationary is also conceivable.

	Wireless Sensor Networks	Mobile Sensing
Hardware	Purpose-built/specialised, complicated (to be programmed at system level)	Off-the-shelf, commodity phones; (+ external sensor units, possibly purpose-built)
Software		Mainstream mobile OSs, frameworks and languages
Networking technology	ZigBee, Bluetooth, proprietary/specialised	Cellular networks (2G/3G/4G), Wi-Fi, Bluetooth
Communication protocols	Ad-hoc, complicated	SMS/MMS, standard Internet protocols (TCP/IP, HTTP, …)
Cost	High	Lower (especially without external sensors)
Potential scale	Limited (due to cost)	Ubiquitous
Spatio-temporal granularity	Sparse (due to cost)	Dense (in urban settings)
Sensing perspective	Stationary/nomadic & *location-centric*	idem (hybrid setups) + Mobile & *people-centric*
Autonomy	High & predictable	Low & unpredictable
Control	High	Low

Table 3.1: Differences between WSNs and mobile sensing systems

tion of making phone calls. Moreover, from a research point of view, various aspects of the sensing process, like accuracy and timeliness, are easier to control in a WSN setup than in a system relying on multi-purpose smartphones – which may disappear in pockets or purses – and their unpredictable, possibly even incompliant, carriers. WSNs remain relevant for many application domains and continue to be the subject of active research. Mobile sensing can be seen as complementary but it also creates entirely new possibilities.

3.3 State of the art

In this section we present a state of the art of mobile sensing research and applications. Apart from contextualising our work, we aim to provide a framework within which we can motivate our choices and contrast them with those of others. First we treat the prototypical architecture of mobile sensing systems. Then, we discuss three dimensions of their design space: application domains, sensing scale and sensing paradigm. We illustrate this with various examples of related work. Throughout the section we frequently refer to a recent survey article by Lane et al. [310], which served as a starting point for this section.

3.3.1 Architecture

Figure 3.1 shows the typical architecture of mobile sensing systems. Most systems follow a *client-server* model in which a client "app" is installed on a number of mobile phones. These clients collect data from built-in or external sensors and communicate through the Internet with a server (or multiple ones) that aggregates all collected data. Depending on the specific system, the clients and server perform several other functions as well.

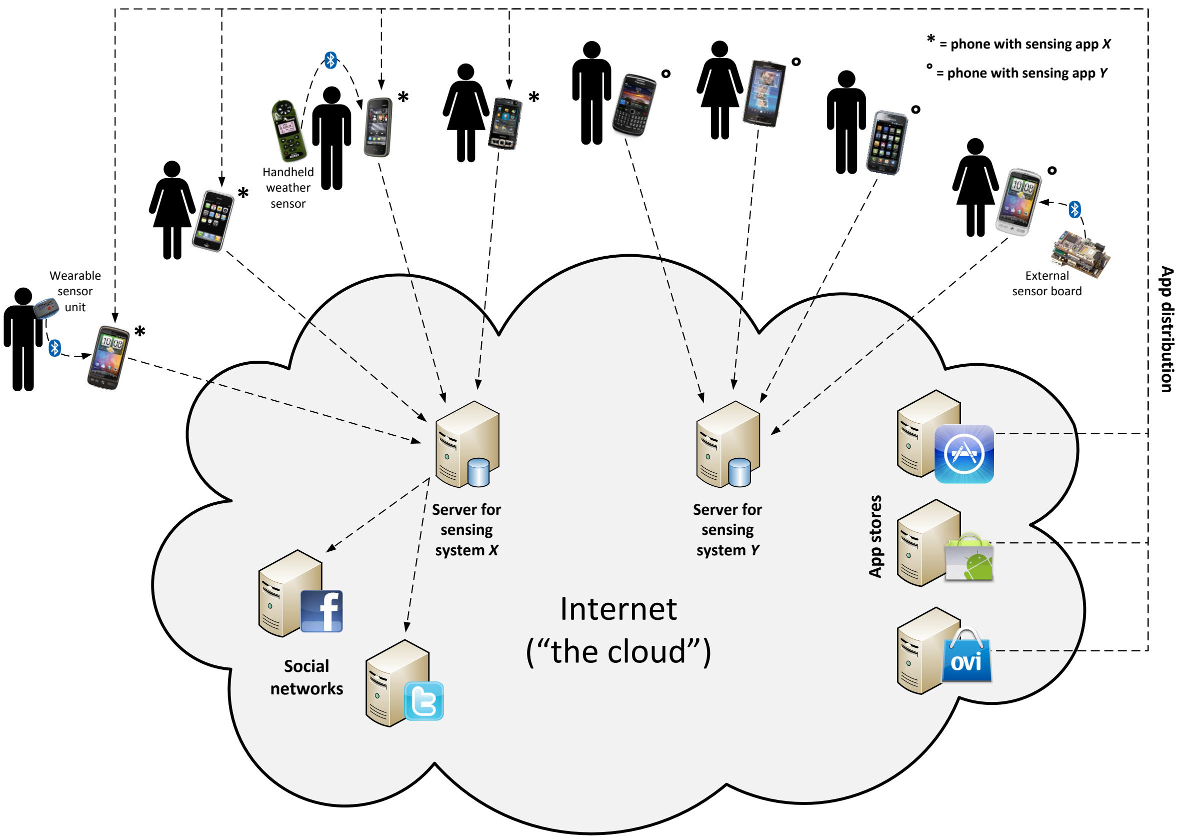


Figure 3.1: Mobile sensing system architecture

Many, but not all, mobile sensing systems meet Doan et al.'s definition of crowdsourcing systems [133] (see section 2.4.3.2): they enlist a crowd of humans, with mobile phones, to solve a problem defined by system creators or owners (which may be the crowd itself).

3.3.1.1 Functional architecture

To discuss mobile (phone) sensing systems from a functional point of view, Lane et al. propose a simple, three-staged architecture, shared by most systems [310]. We generalise by adding the possibility of external sensors and human input. The stages are:

Sensing

Individual mobile phones collect raw sensor data from built-in and possibly external sensors. These include microphones, cameras, light sensors, GPS, accelerometers, gas sensors, etc. In some cases users can act as an additional, human sensor by inputting contextual, semantic or subjective information (e.g. through tagging).

Learning

Raw sensor data, possibly complemented by human input, is processed by machine learning, data mining and other types of algorithms, to extract information on the user's whereabouts, behaviour or various aspects of the surroundings. This interpretation can happen on the phone, on servers ("in the cloud") or both. In the last case, phones pre-process raw data and send a filtered or more abstract representation to a server for further analysis[7]. Where this happens may be dictated by considerations such as privacy, the need to provide real-time feedback, communication costs or bandwidth, available computing resources or the need to combine data from multiple users. In some systems, the user – or multiple users, in a crowdsourcing fashion – can play an active role in this stage, for instance by tagging images [608] or labelling training data for (semi-)supervised learning algorithms [310, 422].

"Closing the loop"

In this stage mobile sensing systems provide feedback loops for the purpose of *informing*, *sharing* and *persuasion*. Individual or groups of users must be informed of the findings that are distilled in the learning stage. This information may be shared with others, possibly after it was filtered or obfuscated (to protect user privacy). To facilitate sharing, information is usually centrally stored on a server[8], where it can be combined with data from other users (aggregation) and from other sources (e.g. weather reports, population statistics, maps, satellite imagery or 3D models). Persuasion takes many forms [187]. Feedback may serve to persuade current users to keep up or increase their contribution and to convince others to join in. It may also serve to persuade users to reach personal goals, such as lowering one's carbon footprint [196] or improving physical fitness [98, 372]. This requires personalised feedback but may also leverage information shared by others. For example, a system could try to motivate or challenge users by presenting comparisons of personal results with those of others. On a grander scale, feedback may also serve to persuade people, users and outsiders alike, of a bigger goal or cause. For instance, an environmental monitoring system could disseminate aggregated data to raise awareness of environmental pollution and possibly even change citizens' behaviour. Depending on the concrete application and the purpose, feedback may take different forms and be delivered through different channels. Often it involves visualisations in the form of graphs, diagrams or maps. Delivery channels include the mobile app itself (immediate feedback to the individual user), the project website, other websites or social networks, and possibly displays installed in public spaces [106].

3.3.1.2 Deployment & scaling

Early, experimental mobile sensing systems were usually only deployed on small scales, for instance in the context of a university campus where researchers self-tested their systems or gave smartphones with a preinstalled sensing app to students or colleagues.

[7] The *split-level classification* applied in *CenceMe* (see below) is an example of such a strategy [351].

[8] In a community memory context, this server is typically the central community memory system itself.

To move beyond small-scale experimentation, some research projects have sought to attract a larger audience by offering a sensing app for download on a website. Nowadays, deployments on a much wider scale are possible, especially for systems that do not require external sensors. There are two main reasons. First, average, not necessarily tech-savvy people have become familiar with the concept of user-installable mobile software. Second, the app store model provides businesses and researchers alike with a – more or less – free distribution channel through which they can reach thousands, if not millions, of potential users from all over the world [310, 349]. Figure 3.1 illustrates this type of app distribution. Recently, businesses have followed academia's lead and have started to develop commercial mobile sensing apps which are distributed through the same channels.

While app stores simplify app distribution, this solves but one of many problems faced by large-scale mobile sensing systems. Recruiting and retaining users remains difficult, since it is not because millions of people can download your app that they will, and it is not because thousands do that they will continue to use it. But even if they do, scaling up the mobile sensing system (as a whole) along with the number of participants remains technically, organisationally and financially challenging. Dealing with potentially massive amounts of data being generated not only requires state of the art algorithms but also extensive infrastructure (e.g. server capacity and network bandwidth). Managing and supporting a community of hundreds or even thousands of needy users, with diverse devices, not only requires user-friendly software but also extensive manpower (e.g. to answer questions, fix bugs, coordinate efforts, etc.). Therefore, Lane et al. rightly ask the question whether academic laboratories are at all geared to scale up their research artefacts from the level of small-scale experimentation to large-scale deployments [310].

3.3.2 Application domains

Mobile sensing systems have been proposed, developed and deployed, by researchers and businesses alike, for a variety of application domains. Below we discuss the main categories, as identified by Lane et al. [310]. For each category we mention some early and more recent examples of concrete systems, also beyond those cited by Lane et al.

Transportation

In urban areas everywhere, issues such as congestion and the organisation of public transportation remain serious challenges. Academics, commercial enterprises and public institutions have (jointly) developed mobile sensing systems to tackle traffic problems [525, 530, 533]. Such systems track the location, speed and destination of their users while they are driving. This data is then used to estimate and predict where and when congestions occur. This enables the systems to provide subscribers with fine-grained, personalised traffic information, accurate arrival time estimates and even real-time alternative route suggestions to help them avoid congestions and to balance traffic to (potentially) reduce congestions for everyone. Aggregated data that is collected in this manner also serves long-term purposes. For instance, companies like TomTom sell anonymised traffic data, partially collected via mobile sensing, to transport and law enforcement authorities, who apply it to advise decisions on construction of new or extended roads, the development of public transport and (until recently) the installation of speed traps [413].

Social networking

The posting of location and activity status messages on social networks has grown popular in recent years. Researchers have investigated how mobile sensing systems can automate and augment this by inferring and classifying the user's current location and activity by applying machine learning algorithms to diverse sensor data collected on his or her phone. The system can then post status messages to social networks on behalf of the users, as illustrated by figure 3.1. An example of such a system is *CenceMe* [348, 349, 351], developed by the Smartphone Sensing Group at Dartmouth College [111] – which is also the affiliation of Lane et al.

Health and Well-being

Mobile sensing systems have the potential to collect in situ data continuously for the purpose of healthcare and well-being, including fitness and sports. One example is the *UbiFit Garden* app, developed by Intel Research and the University of Washington, which captures levels of physical activity and relates this to personal fitness goals. To encourage users, progress towards their goals is visualised on the phone through the metaphor of a blooming garden [98]. Two other research artefacts, *Biketastic* [443, 541], by the Center for Embedded Networked Sensing (CENS) at UCLA, and *BikeNet* [149, 150], of Columbia University and Dartmouth College, are aimed at cyclists. Both systems, which in fact also fall in the categories of transportation and environmental monitoring, enable users to analyse, visualise and share biking routes. Captured parameters include covered distance, speed, calories burned, roughness of the terrain and the noisiness of the surroundings. BikeNet, which employs external sensors, also measures CO_2 levels. A comparable commercial example is *Nike+* [291, 372], by sportswear giant Nike, which uses a phone's built-in sensors, or external ones embedded in running shoes or a watch, to track distance, speed, route and energy expenditure during running workouts. This data can be uploaded to an online platform where runners can analyse their performance, visualise runs on maps, share their accomplishments on social networks and challenge other Nike+ users. The system also encourages users in various ways and can generate personalised training programs. Other applications have been proposed for medical purposes, such as ECG and heart rate monitoring [362, 425].

Environmental monitoring

For our purposes, environmental monitoring is the most interesting domain. Built-in and external sensors allow mobile sensing systems to measure a range of physical parameters that are directly or indirectly related to environmental stressors or pollutants. In urban contexts, mobile sensing systems have the potential to collect massively more, finer-grained data on pollution, and environmental conditions in general, than is possible with fieldwork by professionals or with WSNs. While WSNs are well suited to assess the *emission* or *presence* of pollutants in specific places, the people-centric perspective of mobile sensing systems also allows to assess the *exposure* humans endure during their daily lives and the health or social problems that may cause. Moreover, using cheap, off-the-shelf devices for environmental monitoring enables new scenarios for citizen or community/street science [71, 106].

Pioneering work in this direction was carried out by the CENS group at UCLA [541]. In [2] they presented the *Mobiscopes* concept, a hybrid between vehicular WSNs and mobile sensing systems that could be applied for environmental monitoring. In another position paper, titled *Participatory Sensing* [71][9], the group presented a broader vision for the application of mobile sensing technology in the context of environmental monitoring by citizen scientists. The group also developed a concrete system, the *Personal Environmental Impact Report* (PEIR) [7, 84, 355], which tracks users' whereabouts using GPS and uses those traces to infer transportation modes and various habits. This information is then combined with data on weather conditions, traffic patterns and pollution levels (collected from other sources or simulation models), to create personalised reports that inform users on how their actions affect both their exposure and their contribution to environmental pollution. PEIR also integrates with social networks to let users share and compare results.

In PEIR, and in a similar system called *UbiGreen* [196], no actual pollution measuring happens on phones or external sensors. Others have built systems where this *is* the case. For example, a team at Intel Research and UC Berkeley, which included Eric Paulos[10], experimented with air quality measurement systems based on smartphones with external sensors which were either mounted on a vehicle or carried around by a person [21, 22, 138, 254, 416]. Similar early work was done by researchers from the universities of Nottingham and Cambridge, who developed experimental smartphone apps to assess air quality and allergen levels using external sensors, and noise levels using the built-in microphone [292–294, 420, 421].

NoiseTube, the mobile sensing system that we introduce in chapters 5 and 6 and which forms a core contribution of this dissertation, is situated at the intersection of the environmental monitoring domain and the health and well-being domain.

[9] We already mentioned this paper in in section 2.5.3, and will return to it in section 3.3.4.
[10] Whose work we already mentioned in section 2.5.3.

3.3.3 Sensing scale

Independently of application domains, another dimension within the design space of current and future mobile sensing systems is the *sensing scale*. Lane et al. distinguish three « *distinct scales* » of increasing size: *personal*, *group* and *community sensing* [310]. While we agree that this is a valid and useful way to structure the design space, we want to argue that these scales are *nested* rather than distinct. Moreover, we propose to call the third and largest scale *mass sensing* instead of community sensing.

Personal sensing

These systems are designed to be used by individual users. Here, the app collects data and makes inferences, primarily to inform the user (i.e. the owner or carrier of the device). Information is either not shared at all, or only with a small, user-designated audience, in support of a personal goal, concern or interest of the user. For instance, in applications related to personal health and well-being (see above) information could be shared with a primary care giver, personal trainer or another specialist. Clearly, personal sensing applications are *not* a form of crowdsourcing.

Group sensing

These systems are aimed at groups of individuals that share some common goal, concern or interest. Such groups tend to be formed by individuals that already know each other "outside of the system". Over time, groups may grow as people with similar goals, concerns or interests join existing efforts. An example are sensor-driven social networking applications such as CenceMe, in which status messages can be shared within small, predefined circles of friends and the common interest is of a social nature (e.g. wanting to know where your friends are in order to meet them or join their activities). Other use cases are akin to citizen or community/street science and revolve around a shared scientific interest or local concern. An example of the latter kind is the *GarbageWatch* system, developed by the CENS group at UCLA, which lets groups of students collaborate to gather information on disposal and recycling habits in order to improve the recycling program on their campus [440]. In group sensing applications there is often a level of trust among participants which simplifies otherwise difficult problems, such as protecting user privacy and attesting that data collection happens correctly (or as agreed upon by the group). Many group sensing systems can be seen as a form of crowdsourcing.

Mass sensing

At this scale, massive amounts of people share sensor data and computational resources of their mobile phones, possibly along with manually inputted information, to the benefit of a project or system which usually offers a service in return. People can participate as individuals, or may organise themselves in smaller groups working within the greater whole. Contributing individuals and groups may have diverse motives and can be total strangers to one another who only interact via the system. Of the few operational mass sensing systems in existence, the most successful ones are probably the abovementioned traffic congestion tracking systems [530, 533]. Other potential use cases are tracking the spread of diseases in cities, bird migrations across countries and the creation of urban noise maps (see chapter 5) [310].

As noted above, Lane et al. refer to this level as *community sensing* [310]. However, we believe that the concept of a "community" is too ambiguous in terms of size and may suggest a relationship by place, goal or group membership among contributors, which, while typical for group sensing scenarios, is not necessarily the case here. To avoid this ambiguity and confusion[11] we chose to call it mass sensing instead[12].

Mass sensing systems are prototypical examples of crowdsourcing. The dimensions proposed by Doan et al. [133] to classify crowdsourcing systems on the Web and the identified challenges for their creators (see section 2.4.3.2) are equally relevant in the context of mobile sensing. Arguably the main challenges are recruitment and privacy. Like most crowdsourcing systems, mass sensing projects typically need to recruit a critical mass of users in order to be sustainable. But achieving this implicitly requires the cooperation of strangers who may not trust each other and are therefore much more concerned about their privacy than participants of group sensing projects. Due to the additional challenge of producing sensor equipment on a mass scale and distributing it to participants, or, harder still, convincing them to buy it, scaling up a project involving external sensors to the "mass sensing level" (i.e. hundreds or perhaps even thousands of users) is very difficult. Nevertheless the team from Intel Research and UC Berkeley have envisioned massive, city-wide deployments of air quality monitoring systems. Actual experimentation was limited to a group level and involved custom-made external sensor units, but it is hoped that gas sensors will one day be integrated in off-the-shelf smartphones [254, 416].

Lane et al. state that mass sensing, or community sensing as they call it, systems « *only become useful once they have a large number of people participating* » [310]. We only partially agree with this notion. While it may be true from the perspective of system creators, if it were also true from the users' perspective, then it could be hard, or even impossible, to ever attract a critical mass due to the cold-start problem (see section 2.4.3.1). Based on literature on participant motives in Web 2.0, crowdsourcing and citizen science contexts (see section 2.4.3) and our experience with the NoiseTube system (see chapters 5 to 7), we are convinced that it is essential for creators to ask themselves upfront *when* and *to whom* their system can be useful. To overcome the cold start, it may be necessary, or it can certainly help, to make a mass sensing system useful, but also informative, entertaining, challenging and/or rewarding, on a personal and/or group level as well.

[11] Also with *community* memories, which typically involve group rather than mass sensing (see section 3.5).

[12] Alternatively, we could call it *aggregate sensing*, deriving from the sociological concept of an *aggregate*, which is a cluster of people who, as opposed to a *group*, do not share common interests [553: pp. 89–90].

Similarly, a group sensing system is more likely to be successful in serving the common goal of the group if it is also useful, informative, rewarding or otherwise motivating on a personal level. If sensing scales are seen as distinct classes, there is a risk that cross-scale motivating factors will be ignored or become afterthoughts. Therefore, we claim that it is more appropriate to see the scales as nested levels. Concretely, this means that when developing a larger-scale mobile sensing system it is advisable to create opportunities and incentives for smaller-scale sensing within the same system.

3.3.4 Sensing paradigm

Independently of application domain and scale, mobile sensing systems can be classified by the extent in which the user (i.e. the owner or carrier of the phone) is involved in the sensing activity. According to Lane et al., the two extreme points in this dimension are the *participatory sensing* and the *opportunistic sensing* paradigm [309, 310]:

Participatory sensing

This term was introduced in the eponymous article [71], by Burke et al. of UCLA CENS, which we already mentioned. They characterised the term as follows:

> « *Participatory sensing will task deployed mobile devices to form interactive, participatory sensor networks that enable public and professional users to gather, analyse and share local knowledge.* »

More recently, Estrin, who heads the CENS group, gave this definition [153]:

> « *Participatory sensing is the process whereby individuals and communities use evermore-capable mobile phones and cloud services to collect and analyse systematic data for use in discovery.* »

In participatory sensing systems users consciously and actively engage in the data collection activity by manually determining how, when, what and where to measure. For instance, users may be expected to regularly take the phone out of their pocket or purse to take a photo or measure the ambient noise level. Since participatory sensing was proposed as an ubiquitous computing-enabled version of citizen science [71], participating citizen scientists are thus expected to be willing (and able) to take on a role that goes beyond being passive vehicles for sensors.

Opportunistic sensing

This term was introduced by Campbell et al. of the Smartphone Sensing Group at Dartmouth College. They defined it as follows [74]:

> « [for a] *sensing operation to be successful it is necessary that a* [device]

> *has the right* [sensor(s)]*, is loaded with the appropriate application, and has mobility characteristics that bring the* [sensor(s)] *within the target area during the time window of interest. In an environment* [characterised by] *uncontrolled mobility we term the situation where* [these] *requirements are met as opportunistic sensing.*»

More recently, Lane et al., of the same group, characterised it as follows [309]:

> «*With opportunistic sensing, the custodian may not be aware of active applications. Instead a custodian's device (e.g., cell phone) is utilized whenever its state (e.g., geographic location, body location) matches the requirements of an application. This state is automatically detected; the custodian does not knowingly change the device state for the purpose of meeting the application request. To support symbiosis between the custodian and the system, sensor sampling occurs only if the privacy and transparency needs of the custodian are met.*»

In opportunistic sensing systems data collection is fully automated and requires no involvement of the user (i.e. the device custodian), who participates passively and may even be unaware. Compared to participatory sensing, the burden of sensing decisions is born by the system itself, instead of by the users. Systems must thus be "smart" enough to determine autonomously when to measure and how to interpret the data. This requires apps to be aware of the *phone context* (e.g. whether the device is carried in hand or in a pocket or purse) [310]. Moreover, the needs of the system must be automatically balanced with those of the user. Besides privacy, another user need is that of *transparency*, which implies that the sensing activity should not noticeably impact or interfere with the normal user experience offered by the device (e.g. the primary function of making and receiving phone calls) [309].

Participatory and opportunistic sensing can be seen as the main schools of thought in mobile sensing. Each of the paradigms presents different trade-offs, which we have summarised in table 3.2. Most parameters are related. For instance, the required user commitment, the level of transparency and the likeliness of privacy concerns, all affect how difficult it is to recruit and retain ("R&R") contributors. Similarly, the users' commitment and awareness determines whether there are opportunities for human input, collaboration or coordination. While both paradigms have advantages and disadvantages, it is noteworthy that the drawbacks of opportunistic sensing are largely caused by the fact that, as opposed to participatory sensing, it underutilises a major resource – i.e. people [310].

Note that both paradigms represent extreme points in the design space and concrete systems can exist somewhere in between. For example, an opportunistic/participatory-hybrid system could autonomously collect sensor data, while allowing theuser to provide additional input occasionally, without that being an essential requirement to function.

	Participatory sensing	Opportunistic sensing
User commitment/burden	**Higher** Considerable effort/time/attention required. R&R: -	**Lower** Virtually no effort/time/attention required. R&R: +
Transparency	**Lower** Application may interfere with normal device usage. R&R: -	**Higher** Application works almost unnoticed. R&R: +
Privacy concerns	**Lower** (higher trust) Users are firmly in control of data collection. R&R: +	**Higher** (lower trust) Users may feel spied upon; this can keep or chase some away. R&R: -
Recruitment & Retention (R&R)	**More complicated** (-,-,+)	**Less complicated** (+,+,-)
Main complexity for creators	**Usability** Systems must be informative and engaging to help and motivate users to play an active role, preferably without rending normal usage of their device impossible.	**Algorithmics** Real-time, learning algorithms must autonomously detect sensing opportunities (based on location, time and phone context) while respecting user needs.
"Human sensor" input possible	**Yes**	**No** (in principle)
Supported modes of collaboration	**Implicit and explicit**	**Only implicit**
Central coordination of user efforts feasible	**Yes** (but likely limited by user mobility in space/time) A system could send sensing requests – either automatically or manually triggered by a coordinator – to users who happen to be in an interesting place at an interesting time.	**No** (in principle)

Table 3.2: Trade-offs between participatory and opportunistic sensing

Clearly, the position a new system takes along this "user involvement dimension" has big repercussions on the overall design and the challenges creators face. Such decisions require careful consideration, taking into account the intended goal, the spatio-temporal context, and the motivations, skills and other characteristics of potential contributors.

3.4 Research challenges

Here we discuss the primary research challenges the mobile sensing field is currently addressing or will need to address in the near future. We base this on personal experience and opinion, as well as Lane et al.'s survey article [310] and other publications [106, 416].

Challenge 1: Putting mobile sensing into practice

If mobile sensing is to do its part in the realisation of the ambitious ubiquitous computing and IoT visions (see section 3.2)[13], the field will need to mature and move from experimental to operational deployments. While there remain other challenges to be tackled (see below), we are convinced that today, the field would benefit from a parallel effort which, through applied research, addresses the complexities that impede the transition to operational deployments in specific domains.

[13] As well as our own vision for community memories (see section 2.5).

There are few cases where an operational sensing system could exist as a stand-alone entity – either by filling a void or by completely replacing something else. Instead, to be acceptable, relevant and successful in operational, everyday contexts, most sensing systems will need to be aligned to, or embedded in, diverse existing norms, policies, practices or systems[14]. What that entails depends on the application domain. For instance, sensing systems related to transportation will probably need to interface with various traffic monitoring and management systems currently in use by authorities, and with current channels for the dissemination of traffic information. Similarly, sensing systems in healthcare will need to produce information in specific formats expected by medical professionals. The same is true for environmental monitoring, there too sensing systems may need to complement or integrate with existing approaches (e.g. simulation models) and adapt to prevailing expectations. Clearly, each application domain has specific demands concerning the type, format, presentation, timeliness/recency, and accuracy (see challenge 4) of information. Sensing system creators have little or no control over these demands, yet the degree to which they are met largely determines the acceptance and legitimacy[15] of an operational system. Hence, moving towards operational deployments requires substantial domain expertise. Although part of the process may be a matter of engineering[16], in many cases bridging the gap between theory and practice also requires applied and interdisciplinary research.

Scalability is another typical requirement to be able to move from experimental to operational deployments. As we will discuss below (see challenge 2), recruiting and retaining a large audience of users is deeply non-trivial. Moreover, as we explained in sections 3.3.1.2 and 3.3.3, making the organisational and technical infrastructure scale along may be difficult as well, not to mention expensive. Hence, we must ask ourselves where academia's task should end; or in other words, at which point user communities, authorities and/or commercial enterprises should take over.

To acquire domain expertise and ensure a smooth transition from small-scale experimentation to large-scale operational deployments, we believe researchers – as creators of innovative sensing systems – should seek to forge partnerships with domain experts as well as potential system operators and/or user communities[17]. This means researchers should first and foremost decide *for whom* they create a certain system. For instance, should an environmental monitoring system be targeted at grassroots movements (i.e. bottom-up, citizen-led initiatives), governments looking to increase citizen participation (i.e. top-down, authority-led initiatives), or both?

[14] Note that this applies as much to WSNs as it does to mobile sensing systems.

[15] Especially in the eyes of domain professionals (either academics or officials).

[16] For instance, new and existing systems need to be connected or integrated, requiring the establishment of interfaces and information exchange formats in collaboration with domain experts.

[17] For example, creators of a system for heart monitoring, could (or should) partner with cardiologists' associations, heart patient organisations, and maybe even heart monitoring equipment manufacturers.

Challenge 2: Recruiting & retaining users

Much like Web-based crowdsourcing systems, many mobile sensing systems require a critical mass of users to be recruited and retained in order to succeed in the goals set out by the creators and/or the community of users. This is true by definition for mass sensing systems, but it may also apply to group sensing systems. Depending on the application domain and the sensing paradigm, different strategies may be followed to attract and select contributors – preferably fit for the task at hand – and to keep them motivated, interested, pleased and gratified.

In the context of participatory sensing, some aspects of the recruitment problem have already received considerable attention. For instance, in [440–442] Reddy et al. propose a recruitment and reputation framework for participatory sensing campaigns, which takes inspiration from employee recruitment and human resource management: candidates apply for a job, they are evaluated based on various parameters, their skills are tested through a hands-on assessment and finally the most suitable candidates are recruited; after that, their performance on the job is regularly assessed. In the sensing context, the proposed framework matches campaign requirements (e.g. spatio-temporal coverage) with profiles of candidate participants in order to select the most suitable people to join the campaign, after which their contributions are evaluated by various performance metrics resulting in a reputation value. Participant profiles are mainly based on availability, mobility patterns and habits (captured through annotated location traces), and performance in past campaigns. A shortcoming of this approach is that it assumes that there are enough people willing and able to collect and share their location traces[18], before they are formally recruited to take part in sensing activities [440: p. 144]. This means the first and crucial step of recruitment – finding candidates and convincing them to apply (i.e. collect and share location traces) – is in fact skipped.

Publications on opportunistic sensing seem to ignore routinely the difficulty of recruitment and retention. Instead, they tend to focus on the related but separate challenges of autonomy/transparency, privacy protection and context detection[19]. We believe this is rather short-sighted because, here too, the first step is skipped. Even though users play no active role in the sensing process, they still need to be convinced to install and leave the client app on their device[20]. Convincing enough people to do this, even if privacy concerns would not be an issue, is not trivial.

We are convinced that, regardless of the sensing paradigm, recruitment – especially the aforementioned "first step" – and retention require approaches that are

[18] This typically requires them to install some location tracking application, which many people may be uncomfortable with due to privacy concerns (see challenge 6).

[19] See challenges 6 to 8.

[20] The only alternative is to work with device vendors to preinstall applications on new devices. However, recent scandals have shown that this is a recipe for outrage among privacy-concerned buyers [43].

not purely technical. System creators should foresee and support diverse motivating factors at different levels of scale, including the personal level. After all, the individual motivation to use it is an essential condition for the acceptance of any technology. A lot can be learned from participant motives, competition elements and incentive creation in the scope of Web 2.0, crowdsourcing and citizen science (see section 2.4.3). A promising path is the introduction of monetary incentives to reward and motivate users. Such an approach was tested in the context of participatory sensing by Reddy et al. They conducted a small-scale study involving different micro-payment schemes and found that properly designed schemes have the potential to extend participant coverage both spatially and temporally. Moreover, monetary incentives tended to work best when combined with other motivating factors such as altruism or competitiveness [439].

As noted by Reddy et al., altruism along with other "external" (w.r.t. the system) motivations, like personal interests or concerns, may help to recruit and retain users. For instance, in environmental monitoring campaigns, participants may be self-motivated by a desire to serve their community and concerns about threats it faces. However, that does not mean creators of such systems can ignore this challenge. After all, even militant campaigners may quit if they get too bored or get the feeling their efforts are meaningless. Therefore, it is crucial that these systems provide meaningful, informative, tailored and persuasive feedback (see "Closing the loop" in section 3.3.1.1). In this respect, mobile sensing may benefit from new research into *persuasive technology*, which is a theme within HCI [186, 187].

Challenge 3: Collaboration & coordination

In their survey article on Web-based crowdsourcing systems [133], Doan et al. make a distinction between systems that allow or require explicit collaboration among users, and those that only involve implicit collaboration. The same distinction exists in the sensing stage of mobile sensing systems. Here, implicit collaboration means that each individual user collects data independently (possibly even unaware) of the efforts of others; yet because everyone's data is aggregated by the server, all users implicitly collaborate in serving the project's goals. Explicit collaboration, on the other hand, means that the work is intentionally divided – either beforehand or in real-time – among individual users and/or small, locally collaborating teams.

As indicated in table 3.2, participatory sensing can involve both forms of collaboration, whereas opportunistic sensing only supports implicit collaboration, because, in principle, its users are not expected to take on an active role in the sensing process. Looking at the scale dimension, personal sensing obviously involves no collaboration at all. In group sensing, both implicit and explicit collaboration are possible. Mass sensing relies on implicit collaboration by definition, but may also benefit from the explicit kind. In both group and mass sensing, explicit collaboration has the potential to make data collection more efficient[21] and possibly more enjoyable too. In group sensing, it may arise "naturally" through existing social connections among participants, but nevertheless system creators may want to stimulate it further. In mass sensing, it can probably only occur when somehow stimulated or facilitated by or through the system. Of course, explicit collaboration works best when it is somehow locally or centrally coordinated. Hence, enabling and stimulating explicit collaboration and the coordination thereof, is a challenge for participatory sensing projects at both group and mass scale. Apart from a concern for system creators, this also represents an avenue for (HCI) research, which should devise and evaluate new strategies – both technical and organisational – to stimulate and coordinate explicit collaboration in specific application domains. As we noted in a 2010 paper [499], (semi)-automatic central coordination would be especially useful for group/mass sensing campaigns in the domain of environmental monitoring.

Challenge 4: Data quality

Improving and maintaining data quality is an important challenge for mobile sensing. This applies to the accuracy of measurements and subsequent inferences, as well as the accuracy and representativeness of aggregations. What this means and how important it is for a concrete system depends on its application domain and scale, but also on the goals and ambitions of its creators and/or users. For example, it is less problematic when occasional incorrect inferences of activity or whereabouts cause wrong status messages to be posted to limited circles of social network friends, than when they cause wrong instructions to be sent out to hundreds of drivers – possibly leading to the formation, rather than avoidance, of traffic jams. In his *Designing for Doubt* paper [417], Paulos shares insights into the relative importance of accuracy in environmental monitoring by citizen scientists[22].

One side of this challenge is human. Mobile sensing systems, much like Web-based crowdsourcing systems [133], must balance openness with quality. User behaviour is largely unpredictable and some user actions, or lack thereof, can have detrimental effects on data quality. Usually this happens unintentionally or even unknowingly; for instance due to forgetfulness or a lack of knowledge, skill or time. However, it is also possible that users intentionally act in ways that degrade data quality, misguided by curiosity[23], bias[24], apathy[25] or even malicious intentions[26]. Both opportunistic

[21] E.g. by optimising routes to achieve better spatio-temporal coverage and avoid gaps or double work.

[22] Also see section 2.5.3.

[23] For example, to get familiar with a sensing system and to check if it "works", users often feel the need to "set it off" [417] – e.g. by shouting or whistling in a microphone, or blowing smoke in a gas sensor.

[24] For example, in an environmental monitoring scenario the representativeness of the data can be severely reduced if users only make measurements in places where they personally believe there is a pollution problem and not elsewhere. Worse still, users could try to exaggerate measurements to "make a point".

[25] Lack or loss of interest or motivation can lead to reduced user commitment, effort, compliance and attention to detail.

[26] For instance, some may commit vandalism or deliberate abuse, for whatever reason.

and participatory sensing systems are affected by this problem, yet in different ways. Since users of opportunistic systems are passive in the data collection process, their knowledge, skill, available time and enduring motivation or enthusiasm matter less than in participatory systems [310]. That makes opportunistic systems *less* prone to unintentional human errors or incompliance, but arguably *more* prone to system errors[27]. The fact that participatory systems by design allow users to decide when and where to measure, can make intentional quality degradation easier. However, there are plenty of ways[28] to cause similar disruptions in opportunistic systems. Regardless of the paradigm, system creators are frequently and pertinently asked if their users are at all capable and trustworthy to reliably carry out the task at hand.

The human side of the problem also has human solutions. One is to apply a more selective recruitment policy, that only attracts or allows people who are known to system creators and/or existing users, or who are considered to be skilled and trustworthy. Another is to increase user skill and compliance via training and coaching. However, these solutions may not be scalable, so effectively potential scale is traded for trust. A more scalable solution may be to let the user community regulate itself through (Web-based) moderation, discussion and reputation facilities.

The other side of the challenge is technical. Even if neither user behaviour, nor phone context (see challenge 7), would pose issues, data quality is still limited by the hardware itself. Sensors in mobile phones, or cheap external sensor units, are less accurate than those found in more expensive, more specialised and less portable equipment[29]. This leads to a second, equally common and pertinent, question, namely whether the employed equipment is at all suitable for the task at hand.

Generally speaking, addressing the technical side of the problem requires three steps. First, lab and real-world tests[30] should be conducted to understand and quantify the inherent constraints of the (hardware) platform – i.e. error margins – and how those affect the quality of inferences and aggregations – i.e. propagation of uncertainties. Second, that knowledge should be applied to devise (technical) solutions that allow the system to achieve the best possible accuracy, within those constraints, at the level of individual devices (e.g. correct systematic errors by calibration), and aggregations (e.g. average out random errors). Third, the system's performance should be validated in real-world conditions with real users, eventually also at the intended level of scale. This means system creators should move their project beyond the stage of prototyping and demonstration – which, until now, many fail to do.

[27] For instance, incorrect inference of phone context (see challenge 7) can lead to faulty data.

[28] Some examples are cunning mobility pattern variations, sensing avoidance (e.g. deliberately keeping the phone in a pocket at specific times or places), and even tampering with sensors (e.g. by frequently "setting them off"[23], or even taping them over).

[29] E.g. GPS chipsets in smartphones are less accurate than the DGPS [580] receivers used by surveyors.

[30] Typically through comparison with more accurate equipment.

In response to both sides of the challenge, creators of mobile sensing systems (ourselves included) often claim, or at least hypothesise, that this issue can be addressed by engaging a large and diverse audience of users, who will collect massive amounts of data at different as well as overlapping times and places. The idea is that when a certain scale[31] is reached, it becomes possible to apply statistical techniques to the data[32]. Through averaging and interpolation, outliers and random errors are suppressed and gaps can be filled. Moreover, based on averages and other statistical measures[33], it may be possible to detect suspicious outliers, which can then be removed or given a lower weight – either algorithmically, via human intervention or a combination of both – such that they do not affect the aggregated results in a meaningful way. However, attention should be paid to ensure that only outliers due human or technical errors are deleted or suppressed, and not those caused by rare events[34]. Of course, whether this claim holds in practice depends on the actual scale[31] that is reached. As discussed in challenges 1 and 2, scaling up a mobile sensing system is complicated and may be expensive. Therefore, it may be necessary to focus initially on one or a few localised efforts[35].

Yet even when scaling up the system and the user community is not possible, mobile sensing can still provide useful information, especially if there is no (affordable) alternative (e.g. fieldwork by professionals). After all, often it is likely better to have a set of slightly vague data than no data at all. Moreover, a properly engineered (i.e. tested, calibrated, validated) mobile sensing system should produce data that at least allows relative comparisons between different times and places to be made.

Challenge 5: Reusable components

In recent years, mobile operating systems and application frameworks have evolved at an astonishing pace, which has made mobile app developers' lives a lot easier. Programming smartphones, and cheaper mobile phones too, has become easier thanks to support for popular languages (e.g. Java) and extensive, mobile app development frameworks[36]. Creators of mobile sensing systems have benefited from this evolution as well, mainly because sensing apps share a lot of behaviour and features with other types of mobile apps, especially so-called *location-based services* [47]. Nevertheless, the emergence of reusable, mobile sensing-specific software components – preferably free or open source – could be a significant driver to push the field forward and speed up development of new experimental or operational systems.

[31] Rather than the "sensing scale" (see section 3.3.3), what we mean here is a certain number of users, or, more to the point, a certain amount of collected data per time/place/user (i.e. data density).

[32] Note that it may be necessary to quantify (i.e. calculate) the required scale[31] to reliably apply statistics.

[33] E.g. standard deviation, modus, etc.

[34] Making the distinction between both may require human intervention.

[35] For instance by concentrating recruitment efforts on a (few) neighbourhood(s).

[36] Refer to appendix E for a thorough discussion of these evolutions.

The implementation of a typical mobile sensing system contains a lot of code that, if structured as properly-designed, modular components, could be shared across systems for various purposes and on different mobile phone platforms[37]. Examples of possibly reusable code include low-level sensor sampling routines, context-aware subsystems for duty-cycling and opportunistic sensing[38], learning algorithms, client-server communication APIs, aggregation algorithms, visualisation components, etc. Regarding learning algorithms, Lane et al. argue there is a need for a reusable machine learning toolkit aimed at human behaviour and context modelling in mobile sensing systems, to allow researchers to build and share models on a common foundation. They also point out that the development of new learning algorithms would benefit from public large-scale datasets for training and comparison purposes [310]. Designing reusable components for mobile sensing systems – whether algorithms, datasets, libraries or frameworks – is not trivial. It requires profound insight in the requirements of diverse sensing scenarios – preferably gathered in discussions with system creators and domain experts – extensive knowledge of different mobile phone platforms and a solid background in software design and architecture.

An early, now discontinued, attempt at reuse was the *Campaignr* framework by UCLA's CENS group [82, 289]. The idea was to have a generic mobile sensing app, capable of sampling various sensors, that could be configured to serve as a client for a specific campaign by loading a "campaign file" that specifies which sensors to use, how to capture data, where and how to upload data, and the look of the UI. Campaign initiators would create such files using a simple XML-based format, and distribute them to participants. Participants would then only need to install a single app to contribute to multiple projects, by simply switching between campaign files.

Two more recent attempts are PRISM, short for *Platform for Remote Sensing using Smartphones*, and *EpiCollect*. The former is a reusable platform for building participatory or opportunistic sensing systems, based on a push-based distribution model for sensing jobs and a sandboxed job execution environment for mobile phones [112]. The latter is a framework for the development of participatory sensing apps, and associated Web applications, aimed at epidemiological and ecological data collection campaigns, especially in a citizen or community/street science context [1, 263].

Challenge 6: Privacy

Since most mobile sensing systems involve the collection and transmission of potentially sensitive information (e.g. location traces or speech recordings), the protection of user privacy is very important. After all, if people are uncomfortable with the type of information a system collects, or who may have access to it, they are not likely to (continue to) use it. The data collection needs of the system must thus be

[37] This would be especially useful due to the highly fragmented smartphone market (see section E.2).

[38] See challenges 7 and 8.

balanced with the user's privacy needs. A complicating factor is that privacy preferences are often dependent on spatial, temporal and/or social context [475, 476]. Privacy concerns are especially problematic in mass sensing systems, because there are no or only weak trust relationships among contributors (and system creators).

A simple solution can be to add a pause function in the client app which lets users temporarily interrupt data collection or sharing, whenever they feel uncomfortable with it. Yet that may be insufficient since the resulting data gap could be suspicious in itself. Therefore PEIR applies a technique called *selective hiding*, which fills gaps in location traces with fictitious, yet believable, trajectories that have only limited impact on the quality of the aggregate analysis [355]. Another relatively simple solution is to *obfuscate* data via randomisation or generalisation [61]. For instance, to avoid disclosing a user's precise location, geographic coordinates may be substituted with a neighbourhood or city name. However, obfuscation is not always compatible with system needs[39]. Instead some systems apply *anonymisation*, in which data is stored or aggregated in a way that makes it hard, or impossible, to identify the original submitter(s). More advanced solutions apply cryptographic techniques to encrypt client-server communication and data storage, or to enable *privacy-preserving* distributed computation/aggregation [45, 135, 296, 474]. Privacy concerns can also be reduced by allowing interventions *after* data has been collected and transmitted. For instance users may be allowed to delete, obfuscate, "unshare" or restrict access to parts of their centrally stored contributions [355]. It has even been argued that, to protect user privacy and increase their negotiating power, data collection and data sharing should be decoupled by introducing a *personal data vault*, which securely stores a user's data and from which he/she can then selectively share subsets with various projects or services over time [153].

Privacy protection matters in both participatory and opportunistic sensing, but more so in the latter because users have less control over data gathering. As concerns are often context-dependent, it has been argued participatory systems should actively engage individual users in their own privacy decision-making [153, 475, 476][40]. Opportunistic systems, on the other hand, must autonomously decide when/where to sense and what to share in which (e.g. obfuscated) form, by comparing the current context (see challenge 7) with preconfigured user preferences [295, 309].

Tackling privacy issues takes more than technical measures and promises of well-meaning system creators. Another required ingredient is trust on the part of users. When a user pauses, restricts or disables data collection or sharing, there is always a chance that sensitive information leaks out anyway. One cause could be malicious intentions: system creators could fool users into believing that their wishes

[39] For instance, city-level location traces are fairly useless to track traffic jams.

[40] Just like each individual user plays an active role in the data collection process itself.

are respected while they are in fact deliberately violated[41]. Other, more likely, causes are unintentional bugs or limitations that lead to inadvertent sharing of sensitive information, or exploitable system weaknesses (i.e. security holes). Moreover, the user could have misunderstood the privacy settings, causing more information to be shared than intended. Unless they have certain technical skills, it is often impossible for users to verify personally whether developers respect their wishes, no exploitable bugs are present, and the application is correctly configured. Hence, there is always some level of user trust required. After all, the best privacy protection is not to install *any* sensing app. Building up user trust may require openness and a certain moral or legal accountability on the part of system creators.

Challenge 7: Context-awareness

The topic of context-awareness is a major theme in software engineering, HCI and computer science in general. Notably, Weiser's ubiquitous computing vision [566] relies heavily on this notion. In that scope the term *context* can be defined as [122]:

> « *any information that characterizes a situation related to the interaction between humans, applications, and the surrounding environment.* »

Context-aware software applications can then be defined as [123]:

> « *applications that use context to provide task-relevant information and/or services to a user* »

In the scope of mobile sensing, it can be useful or necessary for client apps to detect and act upon a wide variety of contextual parameters. In the interest of transparency, especially if sensing is continuous (see challenge 8), apps may need to adapt behaviour dynamically in function of parameters related to the device itself (e.g. battery level, incoming/outgoing phone calls, CPU load by other apps, etc.). In systems involving external sensor units (mobile or stationary) that communicate over volatile wireless connections, the presence of units in the surroundings is a dynamically varying parameter in itself: units may come into communication range at any moment in time and may move out of range just as unexpectedly. Client apps must thus respond quickly to the appearance of a unit (i.e. connect to it and query it for data) and be resilient to unexpected, but inevitable, disconnections. Ideally this should happen without user intervention, especially in opportunistic sensing.

Opportunistic sensing systems must autonomously detect and act upon context-dependent sensing opportunities, while always respecting the user's equally context-dependent privacy needs. This requires monitoring user preferences and actions[42] and awareness of the *phone* (*sensing*) *context*, which involves location, travel speed,

[41] E.g. apps could upload data "behind your back" or server-side anonymisation could be a false promise.

[42] For example, phone calls, typing or sudden movements may interfere with measuring.

time and the phone's position with respect to the carrier (e.g. in a pocket, purse or backpack, in the hand, or mounted on the hip or arm) [310, 350]. To detect this last contextual aspect, researchers have proposed machine learning algorithms that combine input from multiple sensors (e.g. microphone, camera or light sensor, accelerometer, gyroscope and compass) [350]. In participatory sensing systems automatic phone context detection is less relevant because users are in charge of detecting and acting upon sensing opportunities[43]. Nevertheless, solutions such as the framework proposed in [350] could be applied to make participatory sensing apps more user-friendly and less prone to faulty data due to forgetful users[44].

Another contextual aspect that can be interesting to capture is the presence of other users (and their devices) in the immediate neighbourhood. This may help to increase (aggregate) accuracy through peer-to-peer self-calibration [255] or super-sampling [256]. Moreover, awareness of nearby users and co-located devices could enable collaborative sensing[45] scenarios [310] in which tasks are locally coordinated or bound to physical places – an idea that has been called *bubble-sensing* [326] – or in which sensors or other resources are dynamically shared, as we proposed in a 2009 position paper [547]. The communication such peer-to-peer interactions would rely on, can be either direct – via local (ad-hoc) networks such as Wi-Fi or Bluetooth, or just plain speech – or indirect – i.e. relayed through the server.

Finding truly robust solutions for context problems in mobile sensing still requires additional research which should build on innovations in machine learning and AI. The field will likely also benefit from advances in software engineering, such as the context-oriented [253] and ambient-oriented programming [115, 116] paradigms.

Challenge 8: Continuous sensing vs. autonomy

Continuous sensing implies that the client application of a mobile sensing system continuously monitors one or more sensors for hours on end, or even as long as the device is switched on. Usually this means the application also analyses raw data in real-time and/or continuously submits data to a server. To enable continuous sensing while maintaining a level of transparency, the device operating system must support multi-tasking and background processes, which is not always the case[46]. More problematic is that continuous sensing puts a heavy strain on device resources: the application may need to frequently sample power-hungy sensors (e.g. GPS); the algorithms used to process raw data may be computationally intensive, causing high energy consumption at the level of the CPU and RAM; and the client may need to communicate frequently with servers over power-hungry connections like 3G/4G.

[43] For instance, users will likely take their phone out of pocket before making measurements.

[44] For example, when a user forgets to stop data collection before putting the phone away in a pocket the app could detect this and interrupt data collection until the phone is taken out again.

[45] Also see challenge 3.

[46] For example, Apple's iOS only recently received (limited) background process support [310].

All of this can lead to drastically reduced autonomy (i.e. battery life), which is a major threat to transparency. This is problematic in general, but especially in opportunistic sensing scenarios, where transparency expectations tend to be higher.

Several approaches have been proposed to tackle this problem. Relatively simple solutions, which may not be universally applicable, include trading off accuracy[47] or responsiveness[48] for reduced energy consumption. Other, more advanced solutions include dynamic offloading of computational work to back-end servers [105], or "smart" *duty-cycling* algorithms, implemented in soft- and/or hardware, which interleave sensor sampling, computation and communication with periods of inactivity [328, 359, 430, 431, 562]. In such solutions a subsystem monitors available resources (e.g. remaining battery capacity), and various other contextual factors (see challenge 7), and makes real-time decisions to balance the system's sensing demands with the user's autonomy and general transparency needs.

Challenge 9: Sensing & learning more

As smartphones get ever more capable, in terms of connectivity and computational power, and the range of built-in sensors grows, mobile sensing will face new opportunities and challenges. Looking at the wide range of innovative sensing applications that have appeared so far, it is unlikely that academics and businesses will run out of ideas anytime soon. New ideas can arise from new sensor types[49] and by using or combining (multiple) existing sensors in new ways. This will enable systems to make inferences about more aspects of our behaviour and surroundings. However, building new inference algorithms, suited for multimodal input and adapted to the challenging environment of mobile, resource-constrained, multipurpose devices, carried by unpredictable, privacy-concerned people, is deeply non-trivial. Realising new opportunities likely requires more research into scalable machine learning or data mining algorithms and distributed computing infrastructures, designed or optimised for mobile sensing. Interesting emerging techniques are the combination of supervised and unsupervised learning in a single system [327] and crowdsourcing the labelling of training data to enable *community-guided learning* [422].

Although most challenges discussed here are interrelated, they are too numerous, diverse and non-trivial to be tackled all at once within a single research project. Instead, researchers, and all creators of mobile sensing systems, should focus on a manageable subset which they consider the most interesting or relevant for a concrete application or domain. As we explain below our work is no exception to this need for prioritisation.

[47] For instance by lowering sensor sampling rate.

[48] For example by delaying processing or communication until the phone is being recharged or until it is connected to Wi-Fi network or to a PC.

[49] For instance, the very latest addition to the range of sensors in smartphones are barometers [317].

Many of the discussed challenges require solutions that are not purely technological. Especially in view of challenge 1, fundamental research must be complemented with applied and interdisciplinary research. Cuff et al. already suggested this as they marked the shift in ubiquitous computing towards urban/mobile sensing in 2008 (see section 3.2) [106]:

> « [...] *this contextual shift* [...] *augurs a fundamental transition from science and engineering into the realms of politics, aesthetics, interpretation, and motivation. More than a change in degree, this is a change in kind that warrants careful, transdisciplinary study.* »

Taking this advice to heart, we will now explain which challenges we have selected to tackle in the research covered in this dissertation.

3.5 Our approach

In section 2.5 we presented our vision for sustainable governance of commons by or with the participation of citizen communities. The principal elements are community memories – as central data repositories and points of interaction for community members and other stakeholders – and the combined use of mobile sensing and social tagging – as a means to collect quantitative and qualitative data about the state of the commons and the health, well-being, behaviour and opinions of those that depend on it. In this chapter, we have so far studied the origins, state of the art and current challenges of mobile sensing as a research field. In this light, we can now refine our vision into a concrete approach, which we follow in the research discussed in this dissertation. In doing so, we position ourselves with respect to the design space of mobile sensing systems, and we identify and motivate the primary challenges we have chosen to focus on in order to realise our vision and contribute to the field.

3.5.1 Mobile sensing systems for commons management

First we map the main aspects of the community memory scenarios, covered in section 2.5.1, onto the framework used in section 3.3 to discuss the state of the art and design space of mobile sensing systems. The goal is to distil a set of specifications for mobile sensing systems in a community memory context.

Architecture

The systems we have in mind follow the typical architecture of mobile sensing systems, discussed in section 3.3.1 and illustrated by figure 3.1. Of course, in this case the server which aggregates data *is* the community memory (CM) system. As mentioned in section 2.5.1, social networks may serve as a dissemination channel, among many others. The typical 3-staged functional architecture is also retained:

- in the ***sensing*** stage community members collect data using a smartphone app, which samples built-in and/or external sensors but also allows users to act as a "human sensor", primarily via social tagging. The use of social tagging in real-world – rather than online – settings, pioneered by Steels & Tisselli [494, 495] and advanced through the research presented here, is not a common trait among mobile sensing systems. However, as motivated in section 2.5.1, it represents a simple yet effective way to gather qualitative information which augments quantitative sensor data in various ways.
- the ***learning*** stage may happen partially in the sensing app, but because general inferences about environmental conditions typically require data about multiple times, places or people, most analysis work would happen on the CM;
- both the sensing app and the CM system should generate various types of feedback to ***"close the loop"***, i.e. to inform, motivate, persuade and support the community and its individual members. This involves real-time displaying of measured values, visualisations such as graphs or maps, etc.

Application domain

Mobile sensing systems for commons management fit in both the **environmental monitoring** and **health and well-being** categories discussed in section 3.3.2. As noted before, early work on sensing systems for these domains by Burke et al. [71] and Paulos et al. [416] has been an inspiration for our vision and approach.

Sensing scale

The characteristics of communities described in section 2.5.1 – i.e. relatively small groups of fellow-citizens with a common goal – align well with the profile of users of **group sensing** systems. However, as argued in section 3.3.3, such systems are more likely to be successful if they are also useful, informative and motivating on an individual level. To facilitate adoption it is also advisable to allow concerned or curious people to collect data on their own without first forming a group of likeminded peers. Hence, we should not ignore the **personal sensing** level. Depending on the initiative taker(s) and the spatial scale, some CM projects may require or attract so many contributors that they approach the **mass sensing** level, in which social connections and trust are weaker and opinions and goals more diverse. To summarise, we want scalable, nested systems, which can be adopted by groups and individuals, who may or may not contribute to larger efforts.

Sensing paradigm

With respect to the sensing paradigms discussed in section 3.3.4, the systems we have in mind follow the **participatory sensing** paradigm, rather than the opportunistic one. This is no surprise since public participation, in all aspects of commons management, is an essential element in the scenarios proposed in section 2.5.1. To benefit from their local knowledge and to raise their awareness through immediate feedback, citizens should be in full control of where and when data is collected, rather than only serve as passive sensor vehicles. While participatory sensing requires more user effort and may be more susceptible to bias (see challenge 4), these issues are outweighed by the advantages.

3.5.2 Primary challenges

As indicated in chapter 1, we intent to conduct applied research which focuses on real societal problems and leads to applicable solutions in the relatively short term (goal 2). Therefore, tackling **challenge 1** – i.e. moving mobile sensing systems from small-scale experimentation to operational, real-world deployments – is our primary research goal. This meant we had to focus on a specific, socially relevant case. Concretely, we have addressed the problem of environmental noise, commonly referred to as noise pollution. To understand the needs of potential users[50] and the expectations of domain experts[51], we have accumulated domain expertise by studying and forging partnerships. This knowledge is summarised in appendix A, which covers the basics of acoustics, human hearing, audio signals, and sound level meters, and chapter 4, which treats subjective aspects of noise, the problem of environmental noise, the response of policymakers, and current assessment methods. We have applied this expertise to design, implement and iteratively improve a mobile sensing and community memory system, true to the requirements set out in section 3.5.1, aimed at monitoring environmental noise and its impact on local communities. This system, called *NoiseTube*, is discussed in chapters 5 and 6. Moving towards applicable solutions requires that we not only develop new technology but also think about possible deployment scenarios. As explained in section 2.5.1, initiatives to set up CMs and data collection campaigns do not necessarily come from citizens, but can also be taken by authorities or academics. Hence, rather than limiting ourselves to a single scenario, we need to take both bottom-up and top-down scenarios into account.

This brings us to **challenge 2**, the recruitment and retention (R&R) of participants, and **challenge 3**, the facilitation of collaboration and coordination (C&C) among them. These are very important in our work, but rather than to develop technological solutions we have focused on organisational ones. As argued above, mobile sensing systems for

[50] I.e. noise-concerned citizens.

[51] E.g. acousticians, environmental policymakers, officials of environmental agencies, etc.

commons management should ideally span multiple, nested levels of scale (personal, group and mass sensing), and initiatives to set them up can be diverse. The scale and type of initiative can affect which specific R&R and C&C strategies may work. In line with our goal to conduct applied research, we have aimed to study, evaluate and improve these strategies by working with actual citizens, as discussed in chapters 5 and 7.

Enabling operational, real-world deployments also requires us to take on **challenge 4** – i.e. the evaluation and improvement of the quality of collected and aggregated data. Although concrete expectations regarding accuracy may differ depending on the scale and type of initiative, it is an important concern for any CM project. We have tackled this challenge in two ways. First, to address the technical side, we have conducted extensive lab tests in collaboration with acousticians. This has enabled us to evaluate and improve (through calibration) the accuracy of noise measurements at the level of individual devices. Moreover, we have developed an innovative solution for the distribution of calibration settings. Second, to address both technical and human aspects of data quality in real-world settings, we have conducted a series of experiments with groups of volunteering citizens. This has enabled us to evaluate the quality of noise maps generated by citizens through participatory sensing at a group scale, and to establish practical guidelines to improve coverage and accuracy. This work is discussed in chapters 6 and 7.

For computer scientists with a strong background in software engineering, designing systems with reusability in mind comes naturally. Therefore, **challenge 5** – i.e. creating reusable software components – is also a priority. As discussed in chapter 6, this has led us to design a cross-platform architecture that enables us to reuse a large portion of program code across multiple mobile phone platforms, which is useful given the rapidly changing market (see section E.2). Moreover, our system has been designed to allow certain components to be reused in other contexts or to allow the existing system to be extended with new functionality – for instance to deal with other types of measurement data (e.g. of air pollution). By releasing the source code under a permissive open source license, we also allow anyone to reuse, extend or improve upon our system.

3.5.3 Secondary challenges

Although we are primarily concerned with challenges 1 to 5, we do not completely ignore the remaining challenges, especially thanks to recent initiatives discussed in chapter 8.

By developing participatory rather than opportunistic sensing systems, **challenge 6** – i.e. dealing with privacy concerns – is less imperative and can be tackled with simpler solutions, because users are in control of where and when data is collected. Hence, while NoiseTube includes a few simple features to increase user trust and control (see chapters 5 and 6), we have initially refrained from focusing on more advanced solutions – typically requiring expertise in cryptography, which is not our specialty. However, as discussed in chapter 8, we have recently collaborated with cryptography specialists to develop an innovative privacy-preserving data aggregation method that could be integrated in future versions of our system [135].

There are many reasons why a mobile sensing app may need to be context-aware. One is to detect automatically when (and when not) data must be collected, which is essential in opportunistic, but much less important in participatory sensing systems. Another is to detect the presence of external sensors, which is not relevant for NoiseTube as it only uses built-in sensors. Hence, while we included some practical features to improve transparency (see chapter 6), **challenge 7** is not a priority. However, as we argued in [547], awareness of, and ad-hoc communication with, other devices and users in the vicinity could enable new ways to collaborate and share resources. As discussed in chapter 8, an initial investigation of this potential was conducted in the scope of a master's thesis [39].

In participatory systems, sensing is only continuous if the user allows it to be. Hence, compared to opportunistic systems, a drained battery is less likely to come as a surprise. Nevertheless, a drastic autonomy reduction can strain adoption of any mobile sensing system. Still, **challenge 8** – i.e. balancing continuous sensing with autonomy – is not a priority for us. Apart from the fact that it is less important in participatory systems, another reason is that tackling this challenge requires expertise in power-efficient hard-, software or protocols, which is not our core competence. However, in chapter 8 we list solutions, proposed by others, which could maybe be applied in future NoiseTube versions.

In order to move mobile sensing into everyday practice (see challenge 1), we have focused on a single case – i.e. environmental noise. Measuring the ambient sound level on a mobile phone can be done through the internal microphone and geo-tagging the data only requires a GPS receiver. Hence, there was no immediate need to sample other sensors. Moreover, rather than to rely on machine learning techniques we leave tasks such as the inference of sound sources to the users (via social tagging). Hence, **challenge 9** – i.e. "sensing & learning more" – is not a priority as such. However, as discussed in chapter 8, this is changing within the *BrusSense* project [124], which aims to extend our vision and approach towards participatory monitoring of air pollution and urban microclimates.

3.6 Conclusion

The goal of this chapter was twofold. On the one hand, it served to situate our work within a particular computer science field and to clarify the origins, current state and open challenges of that field. On the other, it aimed to concretise our approach by contrasting it with the work of others and by identifying the specific challenges we have to tackle.

Our research is situated in the field of mobile sensing. This field – which is also referred to by more or less synonymous terms such as urban sensing, citizen sensing and people-centric sensing – is part of the wider movements of ubiquitous computing and the Internet of Things, and is closely related to the field of wireless sensor networks. After having sketched those origins, we presented a survey[52] of the state of the art, illustrated with examples of related work. This included a discussion of the architecture of mobile sensing systems, deployment and scaling issues, and the main design space dimensions: application domain, sensing scale and sensing paradigm. Next, we listed 9 challenges the field is currently facing. Together this forms a comprehensive introduction to the field, which, apart from contextualising our work, represents a contribution in its own right.

To concretise our vision into a general approach (goal 1), we first laid out a set of specifications for mobile sensing systems in community memory contexts. We did this by mapping the scenarios outlined in section 2.5 onto the framework we used to discuss the architecture and design space of mobile sensing systems. In short, what is required are participatory, multi-scale systems that apply mobile sensing and social tagging to monitor local environmental conditions as well as the health and well-being of citizens. Finally, we outlined the 5 primary challenges to take on in the scope of this dissertation. In line with our ambition to conduct applied research focused on real societal problems, our main challenge is pushing mobile sensing research towards operational, real-world deployments. As this can best be done by concentrating on a specific case (goal 2), we chose to address the problem of environmental noise, which is introduced in the next chapter. The other primary challenges are: the recruitment and retention of participants, the facilitation of collaboration and coordination, the evaluation and improvement of data quality, and the creation of reusable components. Although considered of secondary importance, the remaining 4 challenges – privacy, context-awareness, continuous sensing vs. autonomy, and "sensing & learning more" – have not been ignored. Besides having influenced certain aspects of the technology presented in chapters 5 and 6, they are, or may become, the subject of past, on-going and future work by our group, as discussed in chapter 8.

Our approach – the specifications, and to a lesser extent the prioritisation of challenges – transcends specific commons challenges (goal 1), and is thus relevant for other applied research projects concerned with mobile sensing in the context of commons management and/or citizen science. Therefore we consider it a contribution to the field as well.

[52] This survey builds on, extends and updates earlier work by Lane et al. [310].

Chapter 4

All about noise

4.1 Introduction

Noise is a term that people use to refer to *unwanted* sounds. The labelling of particular sounds as noise, and thereby as unwanted, may be influenced as much, and sometimes more, by personal opinion and contextual and cultural factors, than by physically measurable properties of the sound in question. Despite this subjectivity, there is ample scientific evidence that noise is an increasingly pressing problem around the world. Long-term, excessive exposure is known to have negative effects on human health, well-being and productivity. Hence the notion that noise is an environmental pollutant and a health hazard, rather than just a nuisance, has become widely accepted in recent decades.

In this chapter our goal is to shed light on both sides of the noise phenomenon and at the same time motivate why mobile sensing and social tagging could help to assess its occurrence, dynamics and impact better. First, in order to uncover some of the subtleties and inconsistencies complicating the debate about noise, section 4.2 provides reflections on definitions of noise at the level of individual sounds and/or individual hearers, and draws upon findings from (psycho-)acoustic research. Then in section 4.3 we investigate noise as a societal problem. We focus mainly on the issue of *environmental noise*, commonly referred to as *noise pollution*, the assessment of which is the primary application domain for our research. We discuss the effects of environmental noise on human health and well-being, the response of policymakers (especially in Europe), and current assessment methods and their limitations. After that, section 4.4 concludes the chapter.

For proper understanding of the more technical aspects covered in this chapter it may help to refer to appendix A, which introduces the basics of acoustics, human hearing (including *loudness* perception), audio signals, sound level meters and dosemeters.

4.2 Definitions and perceptions of noise

Establishing a clear definition of what noise *is*, or rather, which sounds are considered *to be* noise, is complicated. When looking at how different individual sounds are perceived by various individual hearers, we quickly find numerous pitfalls, subtleties and complexities. To illustrate this, we paraphrase some dictionary definitions and literary quotations, and then investigate three recurring themes in isolation.

- **Cambridge Dictionary** [72] defines noise as:

  « *(a) sound, especially when it is not wanted, unpleasant or loud* »

- **Oxford Dictionary** [412] defines noise as:

  « *a sound, especially one that is loud or unpleasant or that causes disturbance* »

- **Merriam-Webster's dictionary** [345] defines noise as:

  « [a] *sound, especially: one that lacks agreeable musical quality or is noticeably unpleasant*; [or] *any sound that is undesired or interferes with one's hearing of something* »

- Quotes by **Kurt Tucholsky**, German journalist and writer (1890–1935) [434]:

  « *Lärm ist das Geräusch der anderen.* »

  « *Der eigene Hund macht keinen Lärm – er bellt nur.* »

  « *Es gibt vielerlei Lärme. Aber es gibt nur eine Stille.* »[1]

4.2.1 Unwanted

The definition of noise as "unwanted" or "undesired" sound raises a number of questions. Perhaps most importantly, we should ask ourselves, *"unwanted by whom?"*. Our society is rife with examples of utterly contradictory judgements on physically identical sounds. For generations, parents have been telling their children to *"Turn down that dreadful noise!"*, unable to appreciate the sounds that their offspring thinks have greatly agreeable musical quality. Another example are the lyrical descriptions of the "soundtrack" of powerful sports car or motorcycle engines, as one can find them in motoring magazines and television shows. While motoring journalists go as far as to ascribe musical qualities to engine sounds, those same sounds are generally considered a nuisance by anyone but the drivers of the vehicles in question and other motoring fanatics. As illustrated by the first two

[1] "*Noise is the sound of others.*"; "*One's own dog makes no noise – it just barks.*"; "*There are many noises. But there is only one silence.*"

quotes by Tucholsky, whether or not one considers sounds unwanted, and/or labels them as noise, often depends on the relationship one has to their source. In a recent case, neighbours of a nursery in Bruges, Belgium, went as far as to file a lawsuit over the perceived nuisance caused by the children (presumably none of which were their own), leading to widespread outrage, especially among parents of young children [121]. Clearly, judgements of sounds as being unwanted and/or noise, are highly subjective.

Two other questions we can ask are *"unwanted when?"* and *"unwanted where?"*. Many examples can be found that indicate that the time and place at which a sound is heard can lead to contradictory judgements. At certain times – e.g. at night – or places – e.g. in a church – the sound of a nearby, normal-level conversation, which poses no objection at most other times or places, may be referred to as (unwanted) noise. In many Belgian communes, for example in Leuven [489: sec. 2.3, art. 6 / p. 128], it is forbidden to mow one's lawn on a Sunday, in the evening, or at night. While the sound of a lawnmower is probably considered to be noise by anyone at any time, this sort of regulations do show that the degree of "unwantedness", and thereby our tolerance, of a sound often depends on the time it is heard. Plane spotters, who seem to have no problem with the high sound levels they are exposed to when spotting aircraft during take-off, would probably be as reluctant as anyone (possibly even more so), to go live under the flight path of a major airport. Similarly, even motoring fanatics are unlikely to voluntarily move into a house right next to a racing track, let alone a highway, if they can avoid it. Clearly, apart from subjective opinion, the temporal, spatial and semantic context of sounds can influence whether, or the degree to which, they are unwanted and/or considered noise.

4.2.2 Interfering & disturbing

The characterisation of noise as *interfering* or *disturbing* raises questions about what is being interfered with or disturbed. Again, contradictory judgements are abound. While some people may complain about the loud music they play, that alone rarely causes bars to go out of business for lack of customers. Those who object to it either stay away or leave early, yet plenty of others like, or at least tolerate, loud music and compensate for it by talking louder themselves. In an ironic – or perverse, if you will – effort to create a sense of intimacy, otherwise impossible in a crowded, cramped space, some bar managers deliberately play music so loud that it drowns all conversations except those taking place within a radius of about 1 m or less. In a home setting however, nearby bar-level talking while listening to music, nearby bar-level music while having a conversation, or any nearby bar-level sound while sleeping, usually is an intolerable interference or disturbance to anyone, including avid bar-goers. Clearly, whether sounds are at all found to be interfering or disturbing, and the degree to which interference or disturbance is tolerated, often depends on subjective opinion and/or contextual factors.

The definition of noise as "[a sound that] *interferes with one's hearing of something*" is intriguing for at least two reasons. First, we could ask: if one does *not* want to hear *anything*, does that make *all* sounds noise, or *none*? The answer may depend on what exactly is meant by "not [wanting to hear] anything". If it means "nothing in particular", in the sense that one is indifferent to all sounds, then it could be that none of them are considered (interfering) noise – yet, on other grounds, they could still be considered (unwanted) noise. But if it means "nothing at all", in the sense that one longs for silence – of which there is indeed "only one", as Tucholsky observed – then all sounds are undoubtedly unwanted (and thereby) noise. Either way, this goes to show that there needn't be something that we *want* to hear for (other) sounds to be considered noise. Second, the definition may suggest that what one *does* want to hear is not, or cannot be, noise. We could wonder what this means for people who, like acousticians doing fieldwork or users of our NoiseTube system, measure, study and inevitably listen to what is generally considered as noise. Does it mean that, from their perspective, the labels "noise" and "non-noise" temporarily swap places? Probably not, although an often heard comment among NoiseTube users is that while using it in their neighbourhood, they themselves experience the local soundscape[2] more intensely. To some extent, the act of measuring sound seems to induce people to listen more carefully, causing them to notice sounds, or rather noises, which they otherwise more or less unconsciously ignore. Coming back to the notion of interference, it is remarkable that while measuring noise, or rather sound, one typically tries to be quiet to avoid *interfering* with the measurements. While obvious from a scientific perspective, in a sense this also proves Tucholsky's first observation – i.e. that noise is "the sound of others", rather than of ourselves or all of us.

While our discussion is limited to noise in the acoustical sense, there is a parallel to be drawn to usages of the word in physics, information theory and telecommunications. There too – and this is no coincidence – *noise*[3] is an unwanted, uninteresting or meaningless perturbation that *interferes* with, or otherwise complicates, the detection, measuring or transmission of wanted, interesting or meaningful *signals*, data or information.

4.2.3 Loud and/or unpleasant

Characterising noise as a *loud* or *unpleasant* sound hardly yields objective criteria either. For one thing, *loudness* perception can vary from person to person (see section A.3.4). For another, what (sounds) people find *(un)pleasant* is often – if not by definition – a matter of opinion, taste and context. While this subjectivity does not make the cited

[2] The term *soundscape* was coined in 1969 by Schafer [462] as the auditory equivalent of a landscape – i.e. the whole of sounds and noises that is characteristic for a certain place and/or time.

[3] Unlike English, Dutch has a separate word for this meaning of noise, namely "*ruis*", as opposed to "*lawaai*" which is only used in the acoustical sense.

definitions of noise invalid, we should be extra careful because, besides leading to contradictory judgements, the loudness and (un)pleasantness of a sound are often related.

Inaudible sounds do not bother anyone and quiet sounds are usually easier to ignore than louder ones. Thus it seems only natural that a sound's loudness can significantly affect the degree in which it is considered unwanted, interfering, disturbing or indeed *unpleasant*, and hence referred to as noise. For example, people are more likely to notice and be bothered by the sound of a road 10 m away than of an equally-busy road 100 m away, simply because the former is louder to them. However, that does *not* mean that all noises, or all unpleasant sounds, are necessarily loud – e.g. a dripping tap is relatively quiet but can still be very annoying – *nor* that all loud sounds are necessarily unpleasant or always considered as noise – e.g. music can be loud and enjoyable at the same time.

Although not concerned with the definition of noise per se, numerous psychoacoustic studies have sought to increase our understanding of the complex relation between loudness, pleasantness, annoyance, the *meaning of sound* and various contextual factors. In their comprehensive book on loudness, Florentine et al. note that, while essentially distinct concepts, in practical situations loudness and annoyance are related and easily confounded [182: p. 199]. Generally, the annoyance felt by an individual hearer depends much more on his/her psychological state than on the perceived loudness itself [182: p. 200]. In these studies, the "meaning of sound" is typically taken as a value judgement on its source – i.e. liking, disliking or being indifferent to it. While it may be quite obvious[4] that one's opinion of, or relation to, a sound source can influence ratings of annoyance, or (un)pleasantness, that does not necessarily mean it also affects perceived loudness. Laboratory studies have confirmed that the meaning of sound can influence annoyance, but found little evidence of effects on loudness [182: pp. 200–202]. In daily environments however, there is anecdotal evidence that "non-acoustic parameters", such as neighbourhood problems and personal relationships, can influence both ratings of annoyance and loudness, as well as what level of loudness is considered acceptable [182: pp. 202–203]. Also in daily environments, memories and sensory and cognitive context have been shown to affect annoyance ratings and, to a lesser extent, loudness judgements [182: pp. 217]. Various studies have focused on the role of sensory context, particularly multisensory interactions (i.e. audio–visual and audio–tactile) [182: pp. 208–211]. An especially striking example is the influence of colour. In one study, subjects listened to an engine sound while being shown a still image of a sports car. Except for the colour of the car all images were identical and the sound was played at equal intensity regardless of the car colour. The researchers found that subjects judged the sound as louder when shown a red or dark green car – colours often found on racing and sports cars [344][5]. In another study Zhang and Kang found that environmental factors such as temperature, wind and sunshine can influence the evaluation of soundscapes in urban open spaces [611].

[4] As illustrated above by Tucholsky's dog owners and the dispute between nursery neighbours and parents.

[5] Others had previously identified a similar effect with regard to the colour of trains [182: pp. 209–210].

In fact there is a whole body of research, at the intersection of (psycho-)acoustics, cognitive science and sociology, that is concerned with the evaluation of urban soundscapes, in which loudness and annoyance are just two of many factors. A major topic is the correlation (or lack thereof) between physically measurable acoustic parameters and human interpretation and appraisal of a soundscape (typically assessed using questionnaires). Several studies show that acoustic parameters alone (such as A-weighted sound level, see sections A.2.3 and A.5.1) can only partially explain the perception of soundscapes, which proves the importance of sound source semantics [136, 230, 231, 435, 436]. Other studies, such as [312] by Lavandier and Defréville, focus on building models to predict soundscape evaluations based on acoustical parameters and subjective descriptors.

The case of music is particularly interesting because humans respond differently to it than to other sounds: distortions and certain tonal components, generally perceived as annoying, can be enjoyable in the context of music [182: p. 200]. Similarly, rhythm (and repetition) is an essential aspect of music, but can be a factor of annoyance elsewhere, like in case of that dripping tap. Most importantly, loud music is not necessarily annoying while other loud sounds generally are [182: p. 203]. Here too, loudness and (un)pleasantness are related: loud music can be enjoyable, but loudness can also contribute to enjoyment. This is likely why some people enjoy listening to music at dangerous sound levels, even if they know it can cause hearing loss. Two, possibly complementary, explanations have been put forward: one is that loud music excites the vestibular system (see figure A.3) and is associated with thrill seeking; the other is that some people may develop a behavioural dependency disorder related to loud music listening – i.e. loud music may be addictive. Like other experiences that can lead to behavioural dependency, music can alter moods, reduce pain and elicit craving (i.e. needing to a hear a piece again and again) [182: p. 205]. Personal taste, or "meaning", obviously plays a role as well[6]: an increase in loudness can increase enjoyment for someone who likes the music in question, but can decrease enjoyment, or rather increase annoyance, for someone who dislikes it. Moreover, it has been shown that one's musical taste may also influence judgements of loudness as such. Clearly, « *one person's music is another person's noise* » [182: pp. 207–208].

Whether loud sounds are considered music or noise is of course irrelevant when it comes to the hearing damage risk they pose. Indeed, there is evidence of an increased risk of hearing loss – as well as tinnitus and other hearing damage conditions – among people who frequently listen to loud music on portable media players[7] [109, 160], and among performers, attendees and employees of concert venues and clubs [278, 428, 599, 233]. In a bizarre logic, nowadays concert venues often distribute earplugs, as illustrated by figure 4.1[8]. This way organisers implicitly warn about the potentially dangerous sound level during the concert, while at the same time admitting that they are not willing to do

[6] As do contextual factors, as illustrated above by our comparison of bars and home settings.

[7] These devices are often capable of producing hazardous sound levels; fortunately the European Commission has recently moved to curtail this [336].

[8] Sometimes the earplugs are even sold, which is nothing short of perverse.

Figure 4.1: Earplugs are an essential accessory for concert- and club-goers who are concerned with their hearing

something about it (i.e. lower the sound level). Authorities also bear a responsibility in this matter. In Flanders, for example, until recently the only regulations on the sound level in bars, clubs, concert venues, festivals, etc., were designed to limit the *noise* experienced by neighbours, rather than to protect the hearing of the attending *music* lovers [463].

4.2.4 Summary

The above reflections illustrate that there are many subtleties in the way people define, perceive, interpret and reason about noise. The main conclusion is that, as complex factors related to context, taste and mood come into play, it is often difficult to unequivocally label certain sounds as noise, and thereby as unwanted, interfering, disturbing, unpleasant or intolerable. Hence, making such judgments often requires some form of (internal) debate, but may well be the subject of a permanent difference of opinion.

This goes to show that sound level (i.e. "decibels", see sections A.2.3 and A.5.1) only tells part of the story. We therefore are convinced that, to get insight in the way citizens perceive the soundscape of their daily environment, and to assess the impact of noise on their quality of life, it is useful to collect simultaneously quantitative, physical measurements and qualitative descriptions of sound sources, loudness judgements, subjective perception (e.g. annoying vs. pleasing), spatio-temporal and social context, etc. As explained in sections 2.5.1 and 3.5.1, this is precisely what we hope to enable by combining, in the NoiseTube system, mobile sensing (i.e. letting citizens measure sound level) and social tagging (i.e. letting them comment on what they hear).

4.3 Noise as a societal problem

While noise may be largely a subjective concept, this does not mean it cannot cause objectively observable harm. So far we mostly limited ourselves to the level of individual sounds heard by individual people, now we stand back a little and look at the problem of noise on a collective or societal level.

In both industrialised and developing countries, scientific evidence shows that noise represents a major societal problem. Generally speaking, noise can be problematic where- and whenever the produced sound level (see section A.2.3), the number of sources, the spatio-temporal scale of exposure, the number of exposed individuals and/or the objectively or subjectively observed harm is deemed too high. Typically a distinction is drawn between two categories of noise (problems): *occupational noise* and *environmental noise*.

Within our research we have especially focused on the problem of environmental noise. Therefore we will only briefly touch upon occupational noise. After that we present a thorough discussion of environmental noise.

4.3.1 Occupational noise

Occupational noise is the noise people are confronted with in the context of their job. Problematic exposure to occupational noise may occur in many branches of industry, obvious examples being manufacturing, construction and transport. Excessive exposure to noise in the workplace can cause hearing damage and other health problems (e.g. physical and psychological stress), can interfere with communication and concentration (which in turn leads to accidents and injuries) and reduce productivity [365]. Authorities have established specific regulations to assess and limit occupational noise [174, 365, 544]. In concrete situations, an individual's exposure to occupational noise can be measured with sound level meters or dosemeters. We discuss these devices in detail in section A.5.

4.3.2 Environmental noise

Environmental noise is the noise people are exposed to in their daily lives as a result of various human activities, such as those related to transport, industry and leisure[9]. The picture in figure 4.2 shows a powerful example of environmental noise, in this case caused by transport activity. Depending on the context, other terms, such as *community noise*,

[9] Note that often the same activity can be seen as both a source of occupational noise (i.e. for those who are exposed in their professional capacity) and environmental noise (i.e. for all other exposed people).

residential noise and *domestic noise*, may be used to refer to environmental noise, yet these terms are not necessarily used consistently [572: p. 1].

Figure 4.2: Environmental noise illustration: a Boeing 747-400 aircraft flying low over residential housing while approaching London Heathrow Airport, England, UK (July 2004)

Long-term exposure to excessive environmental noise is known to affect human behaviour, well-being, productivity and health. Environmental noise also has a broader ecological impact, namely on birds and various other forms of wildlife [38, 241, 609]. Ecologists even think that noise pollution may be a factor in some large-scale declines in biodiversity [41]. Especially the human and economic harm it causes, is leading more and more authorities, from international bodies down to local city councils, to recognise, assess and regulate noise as an environmental pollutant that should be reined in where possible.

We begin our discussion on environmental noise with a brief historical, geographic and economic perspective. Then, we explain how exposure to environmental noise is qualified and which thresholds have been established regarding its adverse effects. Next, focusing on the situation in Europe, we look at the main health and quality of life problems caused by excessive exposure, the associated economic and social costs, and the role of inequality. Finally we discuss the policy response from the European Union.

4.3.2.1 Historical, geographical and economic perspective

While complaints about environmental noise have soared in recent decades, as we will discuss below, it is not a new problem at all. Already in ancient Rome, chariots were banned from paved streets at night to prevent the noise of their wheels from disturbing citizens' sleep. Similar measures were taken in some cities in Medieval Europe [210]. The increase of environmental noise is especially associated with technological progress, the development of transport and industry and the process of urbanization [154: p. 2]. Since the late 19th century, the noise produced by new technologies has been the subject of complaints, regulation, legislation and public discussion [51]. The wide recognition of noise as a pollutant, and thereby a potential health risk, is mostly a product of the late 1960s and 70s, as is the term *noise pollution*. Around that time the issue also started to become more prominent on the scientific and political agenda [297: pp. 124–125]. An exemplary statement from that period was made by former U.S. Surgeon General William H. Stewart in 1978 [210]:

> « *Calling noise a nuisance is like calling smog an inconvenience. Noise must be considered a hazard to the health of people everywhere.* »

In the mid-1990s, the European Union (EU) was among the first authorities to make the assessment and regulation of environmental noise a priority. The start of this effort was marked by the 1996 publication of the *Green Paper on Future Noise Policy* [154], in which the European Commission (EC) estimated the extent of the problem and set guidelines for future policies to deal with it. For instance, the EC stated that [154: p. 1a]:

> « *Environmental noise* [...] *is one of the main local environmental problems in Europe and the source of an increasing number of complaints from the public.* »

Based on trends since the 1980s, the EC also indicated that forecasts of the evolution of the problem did not look bright either [154: p. 4].

The problem is equally pressing in other parts of the developed world. In the USA, for instance, environmental noise levels in cities, and complaints about it, have been on the rise for decades [210, 95: pp. 11–17]. Yet American medical professionals still lament that the problem is not being taken seriously enough [210].

The spread of economic development and urbanisation has caused the problem of environmental noise to go global as well, bringing ever more people in harm's way [297: p. 131]. Citizens of rapidly growing agglomerations and booming industrial centres in developing countries (e.g. Mumbai, India [6]) increasingly suffer from and complain about noise. Making matters worse, these countries often have yet to establish laws, regulations and monitoring efforts to curb excess noise.

The link between economic and industrial development – or modernity and prosperity – on the one hand, and *unwanted* sound – i.e. environmental (and occupational) noise – on the other, raises questions about what it is that we, as a society, *do want* (to hear). Since the 1970s, legislation and technological progress in industrialised countries have resulted in significant reductions in the sound level produced by various individual sources – cars, lorries, aircraft and industrial processes are generally a lot quieter now than they were back then – yet there have been little or no improvements in the overall exposure to environmental noise. The reason is simple, due to demographic and economic growth and greater prosperity – leading to increased automobile ownership, tourism and general consumption – road and air traffic, as wells as construction and industrial production, have grown and spread in both space and time, which has largely offset legislative efforts and technological improvements [154: p. 1a].

In the 15 years since the publication of the Green Paper, a considerable amount of data on the exposure of EU citizens to environmental noise and its adverse effects has been collected by EU agencies, the European office of the World Health Organisation (WHO) and national authorities. Below we draw on official reports to sketch a picture of the current extent of the problem within Europe. However, many of the findings – especially regarding caused health problems – are also relevant to other regions.

4.3.2.2 Measures of exposure

Official procedures for the assessment of environmental noise, as well as norms that intend to limit it, generally call for exposure to be measured as the *equivalent continuous sound level* ($L_{\mathrm{eq},T}$; see section A.5.1.3.2) taken over a certain time interval (T) of typically multiple hours. For example, as we explain in section A.5.1.3, a $L_{\mathrm{eq,1h}}$ value signifies the constant sound pressure level (SPL, see section A.2.3) at which a hypothetical sound would, over the course of 1 hour, produce the same amount of acoustic energy during as the measured sound (with fluctuating SPL) did during the 1 hour measuring interval. Environmental noise regulations typically also require that frequency weighting (see section A.5.1.2) is applied to account for the frequency response of human hearing (see section A.3.4.1). Despite the controversy that surrounds it, regulations almost universally call for *A-weighting* to be used. Hence, the measure that is used is A-weighted equivalent continuous sound level, denoted as $L_{\mathrm{Aeq},T}$ and expressed in dB(A).

Official regulations typically specify the periods over which exposure must be assessed. For instance, the EU's Environmental Noise Directive (END) [173], which we discuss further in section 4.3.2.7, stipulates these measures:

$L_{\mathbf{day}}$ is the A-weighted equivalent continuous sound pressure level (L_{Aeq}) over the 12 hour **day** period between 7:00 (7 AM) and 19:00 hours (7 PM).

L_{evening} is the A-weighted equivalent continuous sound pressure level (L_{Aeq}) over the 4 hour **evening** period between 19:00 (7 PM) and 23:00 hours (11 PM).

L_{night} is the A-weighted equivalent continuous sound pressure level (L_{Aeq}) over the 8 hour **night** period between 23:00 (11 PM) and 7:00 hours (7 AM).

L_{den} is the **d**ay-**e**vening-**n**ight level, it is taken over a 24 hour period in which the evening and night hours are emphasised to account for the increased annoyance and disturbance caused during such periods. Concretely, it is a logarithmic composite of L_{day}, L_{evening} and L_{night}, with 5 dB(A) being added to the L_{evening} value and 10 dB(A) being added to the L_{night} value:

$$L_{\text{den}} = 10 \cdot \log_{10} \left(\frac{1}{24} \left[12 \cdot 10^{\frac{L_{\text{day}}}{10}} + 4 \cdot 10^{\frac{L_{\text{evening}}+5}{10}} + 8 \cdot 10^{\frac{L_{\text{night}}+10}{10}} \right] \right) \quad [\text{dB(A)}] \tag{4.1}$$

To account for cultural differences the END allows EU member states to choose other starting hours (but not other durations) for the day, evening and night periods [173].

4.3.2.3 Exposure thresholds

In 1986, the OECD[10] established these thresholds for daytime nuisance, expressed in $L_{\text{Aeq,(daytime)}}$: between 55 and 60 dB(A) annoyance is created; between 60 and 65 dB(A) it increases considerably; and above 65 dB(A) human behaviour patterns are constrained and symptoms of serious health damage arise [401].

In 1999, the WHO advised that, as a precaution against health problems and the disturbance of normal activities of local communities, the L_{Aeq} during the day and the evening (spanning 16 hours) should not exceed 55 dB(A) outdoors, and 50 dB(A) inside dwellings. To avoid sleep disturbance the WHO advised that L_{night} should stay below 45 dB(A) outside, and below 30 dB(A) inside bedrooms [603]. In 2009, the WHO lowered the outside L_{night} guideline to 40 dB(A), but recommended policymakers to aim for 55 dB(A) in situations where 40 dB(A) is not feasible [570: pp. XVII–XVIII].

4.3.2.4 Effects on health & quality of life

Long-term exposure to high levels of environmental noise is known to cause a whole range of discomforts or diseases, often preceded by a long period of annoyance and stress. Based on reports by the WHO and EU institutions, we provide an overview of the main ways in which environmental noise leads to a degradation of health and quality of life. We especially draw upon the 2011 report titled *Burden of disease from environmental noise* [572] by the European WHO office and the EC's Joint Research Centre (JRC).

[10] Organisation for Economic Co-operation and Development.

Annoyance

To date most assessments of the environmental noise problem were based on the annoyance it causes to humans, or the extent to which it disturbs various human activities [572: p. 1]. Annoyance is not only felt because of sleep disturbance or interference with conversations, but also as the less well defined feeling of being disturbed during other kinds of activities as well as during periods of rest [154: p. 28]. Frequent or constant annoyance obviously degrades quality of life, yet – under the WHO's broad definition of health – it is also poses a health risk [572: pp. xvi–xvii].

In 1996, the EC estimated that 17 to 22% of EU citizens (about 80 million people) were exposed – primarily due to road traffic – to continuous daytime outdoor noise levels above 65 dB(A), which is considered to be unacceptable. A further 170 million people were exposed to daytime noise levels of 55–65 dB(A), causing serious annoyance and therefore a reduction of quality of life [154: p. 3].

The degree of annoyance can vary depending on the source of noise. For instance, several national studies in EU countries have indicated that people have a higher tolerance for noise from railways than from road traffic [154: p. 3]. Therefore source-specific noise exposure data is a valuable instrument for policymakers. In accordance with the EU's Environmental Noise Directive [173] (see section 4.3.2.7), this is now becoming available[11]. For instance, data collected between 2002 and 2011 shows that the number of European city dwellers exposed to (potentially) annoying L_{den} levels of > 55 dB(A) is about 55.8 million for road traffic, 6.3 million for railways, 3.3 million for airports and 0.8 million for industrial sites [155: p. 6].

However, due to its subjective nature, annoyance is best also assessed directly, by means of survey techniques such as questionnaires [154: p. 28], instead of only indirectly through estimation of the proportion of the population that is exposed to (potentially) annoying noise levels. The 1995 *Eurobarometer* environmental survey indicated that noise was the fifth most important area of complaints about the local environment, but also the only issue about which the public's complaints had increased since 1992. Moreover, the same survey showed an increase in the public's willingness to take action against noise [154: p. 1]. Looking at the results of the Eurobarometer surveys from 1992 to 2002 [171], we see that noise complaints have indeed been increasing in most EU member states, and in the EU-15 as a whole, as illustrated by chart 4.1. More recent data seems to confirm the trend: compared to the 2002 Eurobarometer [171], Eurofound's 2007 *European Quality of Life Survey* [20] shows a further increase of environmental noise complaints in all EU-15 countries except France and Germany[12], as illustrated by the map in figure 4.3.

[11] But limited to big agglomerations, and major roads, railways and airports outside agglomerations [173].

[12] Direct comparison of both surveys should be taken with some reservation because the questions asked were not exactly the same and there may be differences in the way sample groups were selected.

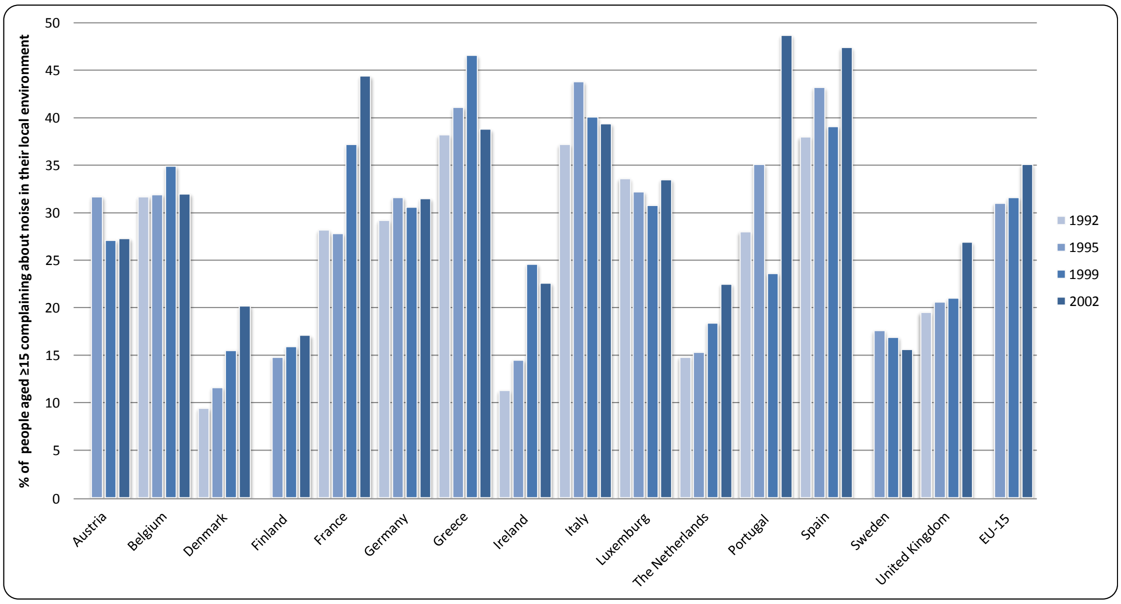


Chart 4.1: Eurobarometer environment survey (1992, 1995, 1999 & 2002): "Percentage of people aged 15 and over who have very much reason or quite a lot of reason to complain about noise in their local environment." [171] (EU-15 countries)

Cardiovascular diseases

Studies suggest that chronic exposure to high levels of environmental noise increases the risk of cardiovascular diseases in the long term. For instance, there is evidence that road traffic noise increases the risk of ischaemic heart diseases, including myocardial infarction (heart attack), but due to a lack of studies it is unclear if there is a similar association with air traffic noise. Yet a growing amount of evidence suggests that both sources of noise increase the risk of hypertension [572: pp. 16–17]. However, the WHO/JRC report states that more research is needed to increase understanding of the relation between environmental noise and cardiovascular diseases. Future research should focus on gender differences [572: p. 30], vulnerable groups, effect modifiers (e.g. personal health history characteristics), hours of the day, coping mechanisms, differences in noise sources, possible confounding with air pollution, difference in objective (sound level) and subjective (sound perception) exposure, and multiple exposures (home, work and leisure) [572: p. 33][13].

Cognitive impairment in children

Studies have shown that chronic exposure to noise negatively affects children's performance in tasks that involve central processing and language, such as reading comprehension, memory and attention. When exposure happens during the critical learning periods at school, it can potentially impair development and have a lifelong effect on educational attainment [572: p. 45]. Current evidence shows that effects can differ depending on the source of noise. For instance, aircraft noise is more harmful than road traffic noise [572: p. 51].

[13] Clearly, this sort of studies will require large, rich datasets to be available.

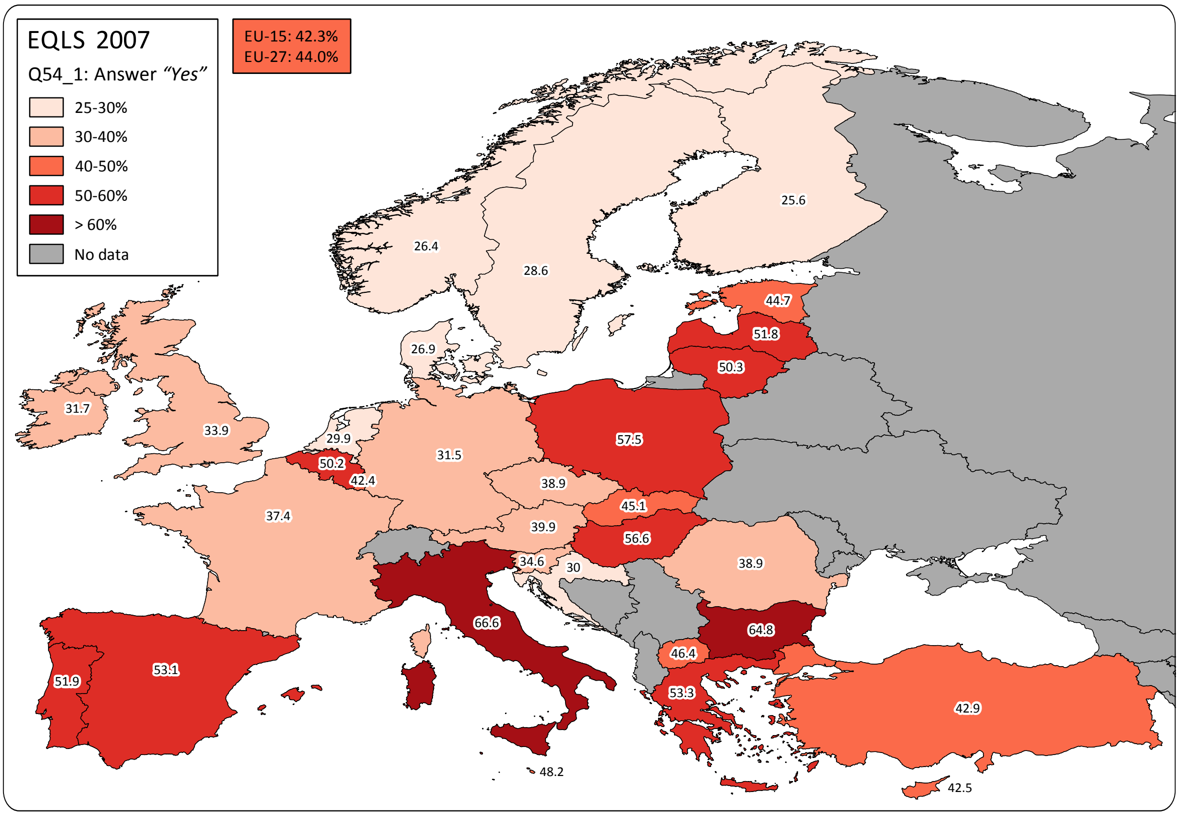


Figure 4.3: A map of Europe showing the per-country percentage of respondents answering *"Yes"* to question Q54_1 of EQLS 2007: *"In the immediate neighbourhood of your home, do you have reason to complain about noise?"* [20, 172]

Sleep disturbance

One of the most common complaints raised with regard to environmental noise is about sleep disturbance. Humans recognise, evaluate and react to environmental sounds even while asleep. Reactions to disturbances during one's sleep can be expressed as changes in sleep structure or increases in heart rate. If such activations happen frequently, this may cause so-called sleep fragmentation which can significantly reduce the restorative power of sleep [572: p. 55]. Continuous noise from 30 dB(A) at the sleepers ear can cause sleep disturbance. Studies have shown that to ensure undisturbed sleep it is especially important that maximum sound levels do not exceed 45 dB(A) [154: p. 27]. Yet even indoors that level can be easily surpassed when there is heavy road traffic nearby. Apart from causing annoyance and complaints, sleep disturbance can also have a major impact on daytime performance and general health. During the day people may experience a deterioration in mood or symptoms like tiredness, headaches and a nervous stomach [154: p. 27]. Acute and chronic sleep restriction and fragmentation due to disturbance affect psychomotor performance, memory consolidation, creativity, risk taking behaviour and signal detection, and thereby increase the risk of accidents [572: p. 55].

Tinnitus & Hearing loss

Tinnitus is defined as the sensation of sound, sometimes called *phantom sounds*, that cannot be attributed to actual external sound. It commonly coincides with hearing loss, but not always. Cases of tinnitus differ in the duration of single episodes (intermittent for seconds or minutes at the time, or continuous), duration of the overall condition (days, months, years) and severity (degree of annoyance, interference with daily life). The condition can cause sleep disturbance, cognitive effects, anxiety, psychological distress, depression[14], communication and listening problems, frustration, irritability, tension, inability to work, reduced efficiency and/or restricted participation in social life [572: pp. 71–72].

Excessive exposure to noise is an important cause of tinnitus: 50 to 90% of patients with chronic noise trauma report tinnitus. Moreover, between 12 and 50% of people suffering from noise-induced hearing loss (NIHL) also report having tinnitus. The WHO notes that, in the majority of the population, no NIHL is to be expected at occupational noise levels of $L_{\mathrm{Aeq,8h}} < 75\,\mathrm{dB(A)}$ and environmental noise levels of $L_{\mathrm{Aeq,24h}} < 70\,\mathrm{dB(A)}$. While tinnitus does not always coincide with NIHL, the WHO considers it reasonable to use the same protective levels for tinnitus [572: p. 73].

While the studies cited in the WHO report focus on the relation between tinnitus and environmental or occupational noise, this and other exposure-induced hearing damage conditions can just as well be caused by sounds which the listener, and future patient, did not consider as "noise" at all, as we discussed in section 4.2.3.

In the 2011 WHO/JRC report [572], the *estimated burden of disease* caused by environmental noise is quantified using the *disability-adjusted life year* (DALY) indicator. This standard disease burden measure represents the total number of "healthy" years lost, per year across a population, due to ill-health, disability or early death. Table 4.1 lists the results for each of the abovementioned problems caused by environmental noise.

	Disability-Adjusted Life Years (DALYs) lost
Ischaemic heart disease	61000
Cognitive impairment of children	45000
Sleep disturbance	903000
Tinnitus	22000
Annoyance	645000
Combined	1.0–1.6 million

Table 4.1: Estimated burden of disease from environmental noise in western Europe [572: p. xvii]

[14] There are even reports of suicide [572: p. 72].

It goes without saying that the accuracy of such risk assessments depends on the availability and quality of data. In this respect the WHO/JRC report points out a number of issues, notably the fact that, while EU member states have started to systematically assess exposure to environmental noise in large agglomerations[15], population-wide exposure data – notably for less densely populated rural areas – is still largely missing. Indeed, in order to estimate disease burden across western Europe[16], the authors have often resorted to extrapolation of urban exposure data. Even though they have tried to avoid overestimation by taking conservative assumptions[17], the authors do state that population-wide data is desirable [572: pp. 7–12, 33 & 67]. Hence, it is clear that in order to increase our knowledge about – and eventually limit or reduce – the impact of environmental noise on the health and quality of life of specific local communities, as well as entire populations, much more, detailed, recent, source-specific exposure and survey data is needed. As we will explain in chapter 5, we believe the combination of community memories, mobile sensing and social tagging can form an excellent platform to collect massively more data on environmental noise exposure and the (perceived) harm it causes.

Although more data is needed, the available information leaves little doubt on the gravity of this problem in Europe. In a 2010 report titled *Health and Environment in Europe*, the WHO stated that environmental noise is considered the most common environmental health stressor, affecting a quarter of the EU population [571: p. 67]. According to the 2011 WHO/JRC report, every year this leads to an estimated total of up to 1.6 million healthy years lost in western Europe [572: p. xvii]. Moreover, current trends provide little hope for improvement: while exposure to other stressors (e.g. second hand smoke, dioxins and benzene) is declining, exposure to environmental noise is still increasing [572: p. 1].

4.3.2.5 Economic and social costs

Due to its impact on health and quality of life, environmental noise also creates considerable economic and social costs. Regarding these costs, the EC stated in 2011 [155: p. 2]:

> « *Economic costs of noise pollution include devaluation in house prices, productivity losses from health related impacts and distributional impacts. Social costs are related to premature death or morbidity (poor concentration, fatigue, hearing problems). The social costs of traffic, rail and road noise across the EU was recently estimated amount to € 40 billion a year, of which 90% is related to passenger cars and goods vehicles. This was about 0.4% of total EU GDP including health care costs.* »

[15] In accordance with the Environmental Noise Directive [173] of 2002 (see section 4.3.2.7).

[16] Due to a lack of exposure data in south-east Europe and newly independent states, the authors have not been able to estimate disease burden for the entire European continent [572: p. xvii].

[17] This is also needed due to the many uncertainties that are inherent to the methods that are currently used by authorities to assess exposure to noise [572: pp. 3–7] (see section 4.3.2.8).

4.3.2.6 Environmental noise and inequality

It has been suggested that the risk of being exposed to hazardous environmental noise levels tends to be higher for those in a weaker socio-economical position [297: pp. 4–9]. To see if this claim holds in the European context, we take another look at results of the 2007 European Quality of Life Survey [20, 172]. Table 4.2 shows the respondents' appreciation of environmental noise as a reason for complaint in function of income.

EQLS 2007, Q54_1: Answer *"Yes"*, %	Lowest income	Low income	High income	Highest income	Overall
Belgium	**57,1**	51,5	49,8	40,4	50,2
Bulgaria	53,5	57,4	57,0	**70,0**	64,8
France	39,1	39,9	**41,4**	37,1	37,4
Germany	**41,4**	28,0	31,1	28,1	31,5
Italy	72,3	66,0	**72,6**	58,7	66,6
Spain	**65,2**	51,4	51,3	47,0	53,1
United Kingdom	37,4	36,8	38,6	**40,6**	33,9
EU-15	**45,3**	39,7	41,1	37,7	42,3
EU-27	**45,0**	40,6	43,8	41,9	44,0

Table 4.2: Percentage of respondents per income category, in a selection of countries, answering *"Yes"* to question Q54_1 of EQLS 2007: *"In the immediate neighbourhood of your home, do you have reason to complain about noise?"* [20, 172]

Interpreting these results is not straightforward. When only looking at the extremes, they seem to confirm the role of income inequality. In the EU as a whole, and especially in the EU-15 countries, more people in the lowest income category felt entitled to complain than in the highest category. In countries like Belgium, Germany and Spain that difference was even more pronounced. However, the effect of income is not always very strong, nor is the trend always consistent across the income range. In some countries, like Bulgaria and the UK, we even see the reverse. This does not necessarily negate (nor does it prove) the role of income inequality in exposure to environmental noise. One hypothesis to explain the results is that the tendency to complain about noise is not only related to the gravity of exposure itself, but also to the expectations people cherish (e.g. regarding the standard of housing) and their political and social capital in general.

We should note that the results of the EQLS survey are aggregated on a country-level, which makes it impossible to account for the kind of places – e.g. urban vs. rural, proximity of busy roads, airports or industrial concentrations, etc. – in which people live. Again we see there is a need for more, richer datasets. We are convinced that community memory initiatives, which use citizen-collected data – not only about exposure and complaints, but also about various socio-economical parameters and other local knowledge – to produce detailed maps about local conditions, can help to answer this sort of questions.

4.3.2.7 Policy response

With the 1996 *Green Paper on Future Noise Policy* [154] the European Commission moved the issue of environmental noise higher on the EU's political agenda. The next big step was taken on 25 June 2002, when the European Parliament and Council adopted Directive 2002/49/EC, better known as the *Environmental Noise Directive* (END) [173]. Even before 2002, some EU member states had already been producing noise maps of densely populated areas. The END has made that obligatory for all member states and has, to some extent, standardised the type of data and maps that must be produced. This has spurred further development of noise assessment methods, and has led to an inventory of noise maps of large cities throughout Europe. Hence, the EU is now at the forefront when it comes to the assessment – but not the reduction – of environmental noise [290]. Before discussing the concrete requirements of the END, we take a brief look at the legislative situation in other countries.

The United States, though its *Noise Control Act* dates back to 1972 [97], does not mandate noise mapping at a federal or state level. Hence it has been slower to catch on to regional noise mapping, and community-wide noise maps are rare. Nevertheless, the same methods that are applied in Europe could be implemented in the U.S., and such efforts are indeed underway [290]. The situation for other developed countries, such as Canada or Japan, is typically similar to that of the U.S., while most developing countries still lack national legislation for noise assessment and noise control.

The purpose of the END is to « *define a common approach intended to avoid, prevent or reduce on a prioritised basis the harmful effects, including annoyance, due to the exposure to environmental noise* ». To achieve this, it requires member states to: determine the exposure to environmental noise through noise mapping; make this information available to the public; and adopt action plans based on the noise mapping results [173: Art. 1]. Concretely, they must ensure the following data is collected [173: Art. 7–8 & Annex VI]:

- For agglomerations with a population of > 250,000, no later than June 30, 2007; and for those with a population of > 100,000, no later than June 30, 2012:
  - the estimated number of citizens living in dwellings that are exposed to L_{den} values of 55 to 75 dB, in bands of 5 dB, and over 75 dB, at 4 meters above the ground on the most exposed facade. Separate estimates are required for road, rail and air traffic and for industrial sources.
  - the estimated number of citizens living in dwellings that are exposed to L_{night} values of 50 to 70 dB, in bands of 5 dB, and over 70 dB, at 4 meters above the ground on the most exposed facade. Separate estimates are required for road, rail and air traffic and for industrial sources.
  - graphical presentation of this data on strategic noise maps, which must show at least the 60, 65, 70 and 75 dB contours.

For agglomerations with a population of $>250{,}000$, no later than July 18, 2008:

- an action plan, designed to manage noise issues and effects, including noise reduction if necessary, and to protect quiet areas against an increase in noise.

- For major roads (>6 million vehicle passages per year), major railroads ($>60{,}000$ train passages per year) and major airports ($>50{,}000$ movements[18] per year), no later than June 30, 2007:
  - the estimated number of citizens living outside agglomerations in dwellings that are exposed to L_{den} values of 55 to 75 dB, in bands of 5 dB, and over 75 dB, at 4 meters above the ground on the most exposed facade.
  - the estimated number of citizens living outside agglomerations in dwellings that are exposed to L_{night} values of 50 to 70 dB, in bands of 5 dB, and over 70 dB, at 4 meters above the ground on the most exposed facade.
  - the total area (in km^2) exposed to L_{den} values higher than 55, 65 and 75 dB and the estimated number of dwellings and inhabitants of each of those areas. The 55 and 65 dB contours must also be shown on one or more maps.

  No later than July 18, 2008:

  - an action plan, designed to manage noise issues and effects, including noise reduction if necessary.

All datasets, maps and action plans must be revised (at least) every 5 years[19].

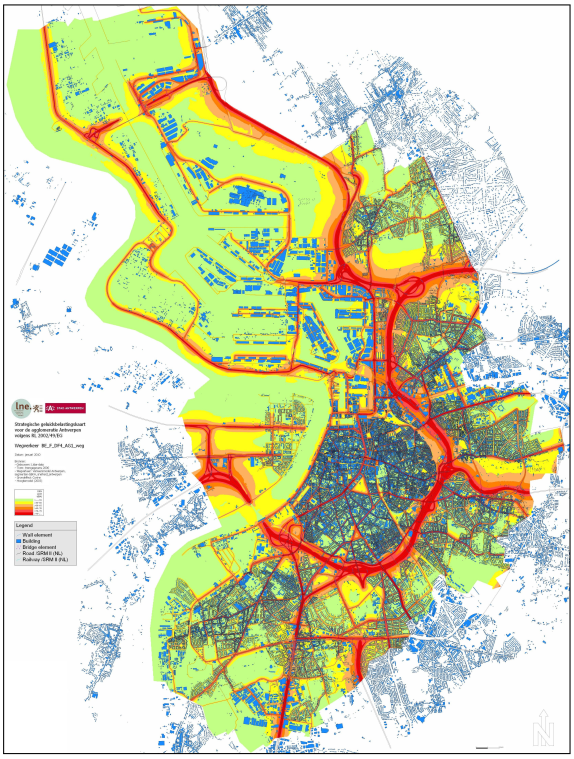
The most visible part of the implementation of the directive is the creation of strategic noise maps. These show L_{den} and L_{night} values for an average day/night in an indicated year as colour-coded bands of 5 dB. Separate maps must be made for road, rail and air traffic and industry-related sources of noise. In figure 4.4 we see three examples of such maps, for road traffic noise in the cities of Brussels, Antwerp and Paris. As is clear from these examples, the END does not specify a standardised colour scale. Consequently the maps for different cities, even within single countries, often use different colours, complicating visual comparison. Though not mandatory, some cities also provide a map that shows the combined exposure for all considered sources. What makes these maps "strategic" is their long-term focus. They are valid for 5 years and represent the average sound level one can expect for a limited number of environmental noise sources. Hence, they do not cover individual, incidental, local or short-term events (e.g. roadworks, sirens, noisy neighbours, etc.) nor occupational noise.

Thanks to the END a lot of data about long-term exposure to environmental noise in EU agglomerations is becoming available. However, as noted above, there is still a shortage of

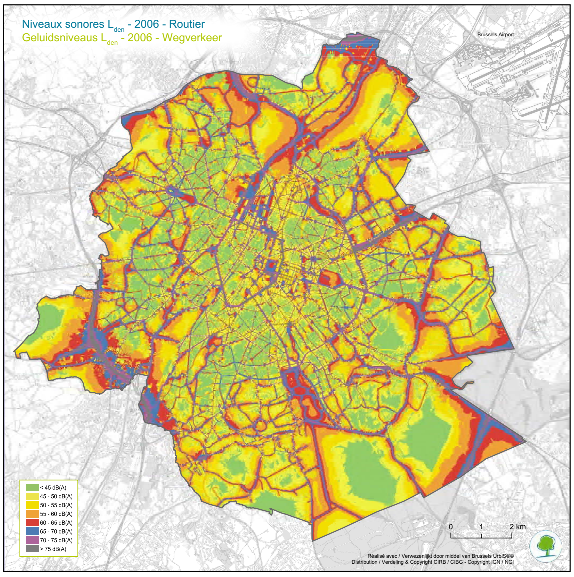
[18] A movement being a take-off or a landing.

[19] Some of the abovementioned deadlines may have been revised since the adoption of the END, an up-to-date list can be found in [167].

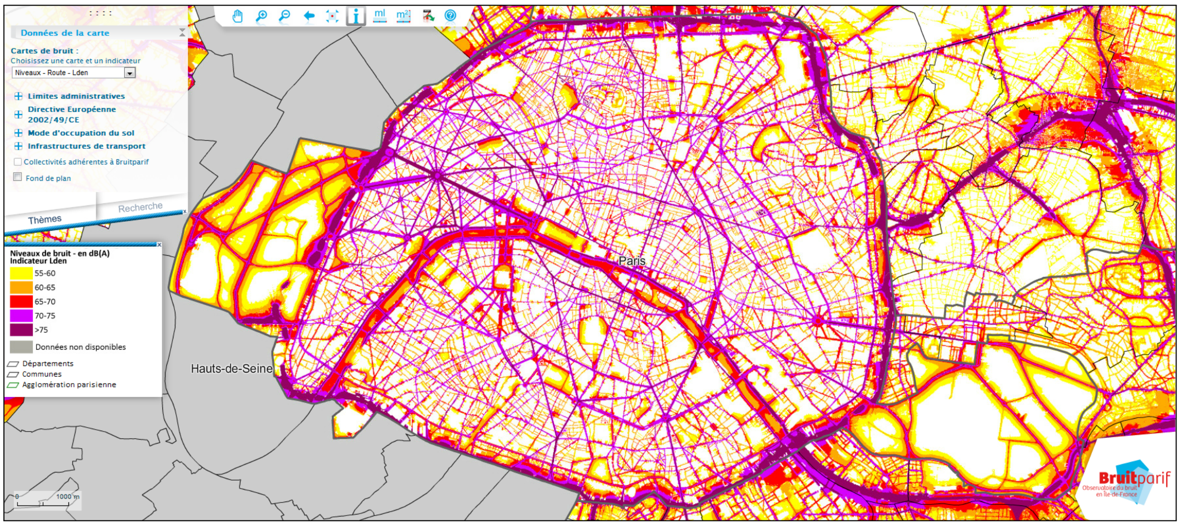
data about less densely populated areas and about local, short-term variations anywhere. Moreover it remains to be seen whether the END action plans, which are now being enacted, will result in significant reductions of environmental noise exposure. In other words, while local and national authorities may have a better idea of where and when the WHO exposure thresholds [570: pp. XVII–XVIII] are likely to be surpassed, coming up with policies to do something about it may still turn out to be difficult.



(a) Official L_{den} map for road traffic noise in Antwerp, Belgium (2010) [557]



(b) Official L_{den} map for road traffic noise in Brussels, Belgium (2006) [64]



(c) Official L_{den} map for road traffic noise in Paris, France (2007) [60]

Figure 4.4: Examples of END-compliant noise maps for three European cities, generated using simulation models

4.3.2.8 Current assessment methods

We now discuss the methods that are commonly used to assess urban environmental noise systematically. Again it is important to consider the role of the END, as this legislation has been a principal driver for the development of at least one of these methods.

4.3.2.8.1 Simulation models

The END states that [173: Annex II]:

> « *the values of L_{den} and L_{night} can be determined either by computation or by measurement* »

In practise, however, the exposure values on END-compliant noise maps, such as those in figure 4.4, are virtually always computed. The main reason is scalability: authorities have hitherto considered it infeasible to measure the sound level at all places and times. Hence most, if not all, END-maps are produced with specialised software that simulates (i.e. computes) expected levels at different places and times. Such software uses source-specific sound propagation models that are fed with statistics about the presence of considered source(s) – e.g. the average number of vehicles on roads, the frequency of planes on low-altitude flight paths, etc. – and information on urban topology – e.g. the height of buildings, the width and surface type of roads, the presence of noise barriers, etc.

Even though measuring is allowed the END implicitly assumes modelling. For instance, the requirement to make separate maps per sound source is difficult to fulfil with measuring since sound level meters (see section A.5.1) cannot differentiate between sources. The preference for computation was made explicit in the *Good Practice Guide for Strategic Noise Mapping and the Production of Associated Data on Noise Exposure* by the EC's Working Group on the Assessment of Exposure to Noise (WG-AEN) [159: p. 10][20]:

> « *WG-AEN recommends that wherever possible strategic noise mapping should generally be carried out by computation. However, it is recognised that noise measurement has many supplementary roles to play in the effective implementation of the END.* »

Two such "supplementary roles" for measurements are the initialisation of models and the verification (and possibly the correction) of their output. Typically this only requires limited amounts of data which is either collected by designated officials equipped with professional-grade sound level meters, or by means of a sensor network (see below).

[20] While the « *WG-AEN strongly recommends that every effort should be made to obtain* ***accurate real data*** *on noise sources* » [159: p. 7], that applies to statistics (e.g. $\frac{\text{cars}}{\text{hour}}$), rather than L_{Aeq} measurements.

Advantages

The main advantage of simulation is that it allows to predict sound levels over huge areas with little or no measuring in the field – provided that statistics on considered sources are available.

Limitations

High cost

Applying this method requires large datasets (which may not be publicly available), expensive software, powerful computers and a lot of human expertise. Therefore many local or regional authorities subcontract the creation of strategic noise maps to specialised firms. Although cost estimates are hard to come by it is clear that this method is very expensive. Hence it may be out of reach for NGOs and even for some authorities (e.g. in developing countries).

Uncertainty

Because the output of the models is only validated with a limited amount of measurements, it is hard to estimate overall error margins. Moreover, accurate modelling of sound sources requires many different input parameters, all of which can influence the accuracy of the result[21]. For instance, to model road traffic, one needs to account for the frequency of different vehicle types, road surface types, weather conditions, the dimensions and surface material of nearby buildings, etc.

Limited amount of sources

In reality the diversity of sources of potentially bothersome noise is much broader than the 4 sources considered by the END (road, rail and air traffic and industry). Hence, as noted before, END-compliant maps ignore the noise produced by humans (e.g. noisy neighbours), construction sites and roadworks, passage of emergency services, unscheduled flights (e.g. helicopters), various types of manifestations or events, etc. – all of which can be highly annoying to at least some city dwellers. Moreover, even for sources which *are* being considered far from all instances are included. For instance, for road traffic maps only roads with >6 million vehicle passages per year must be modelled. Of course this is partially a matter of policy. Indeed, future END revisions could oblige member states to take additional sources into account and bring down inclusion limits (i.e. also model less busy roads). Yet to some extent this limitation is inherent to the method itself. Considering additional sources may require new data to be collected and new propagation models to be developed, and may significantly increase the computational workload – all of which would likely drive up costs, possibly beyond what is affordable.

[21] Insights in different sources of uncertainty in noise modelling are discussed in [159: Appendix 4].

Strategic focus

END-compliant noise maps are valid for 5 years and represent expected sound levels for an average day or night. Hence they do not inform about incidental or short-term variations in exposure. For instance, if road traffic is deviated due to roadworks or a manifestation this can significantly alter the soundscape of a neighbourhood for days, weeks or even months, but it will not be reflected in the maps. Again this is not solely a matter of policy but also a practical limitation of the method itself.

Simulated exposure vs. actual discomfort

As discussed in section 4.2 the perception of noise and the annoyance it causes is highly dependent on diverse contextual factors, which are not taken into account in simulation models nor represented on noise maps. Moreover the simulations do not model indoor noise levels, even though the majority of people spend most of their time there. Hence the statistics on the number of dwellings and inhabitants exposed to certain sound levels, which the END requires member states to submit to the EC, are based on simulated outdoor levels. Apart from being influenced by what happens outside, indoor sound levels also depend on architectural aspects (e.g. orientation, insulation and glazing) and of course the presence of indoor noise sources. By ignoring contextual factors and only modelling outdoor noise it is possible that incorrect conclusions are drawn regarding the actual discomfort.

Simulated maps are useful for authorities to understand the global trends in the urban soundscape, providing a lower limit on actual citizen exposure. They do not, however, capture person-centric exposure levels, nor do they model local variations very well. Therefore they have remained of little interest to citizens, who either perceive the obvious (e.g. their street is noisy), or, worse, cannot link the patterns on the map to their own experience (e.g. their street is not modelled, local or short-term variations are ignored).

In [356] Murphy & King provide a more thorough critique of the strategic noise maps required by the END and the computation methods advised by the WG-AEN. They also make recommendations for future amendments to the legislation.

4.3.2.8.2 Sensor networks

Another assessment method that is increasingly used by local and regional authorities – at least in Europe – is based on sensor networks. These consist of nodes that are installed at various locations across a city and which autonomously and continuously measure the ambient sound level. Each such node can be either stationary (i.e. permanently installed, often on rooftops or street furniture), mobile (i.e. installed on a vehicle) or nomadic, which means that it is regularly moved to a new location. Typically authorities try to distribute the nodes in such a way that the network covers different representative areas, affected by different dominant sources of noise (e.g. near roads, railways, industry, etc.).

While the equipment used to build these networks is not necessarily wireless, nor battery-powered, this approach is in many ways similar to the wireless sensor networks (WSNs) we discussed in section 3.2.1.

For instance, BIM/IBGE, the environmental agency for the Brussels Region, uses a network consisting of 17 stationary nodes [63]. Another example exists in Paris and the surrounding Île-de-France region, where the BruitParif agency has set up the *Réseau RUMEUR*, consisting of 20–30 stationary and nomadic nodes [58]. Both agencies provide a Web portal, respectively [62] and [59], through which citizens can consult the data that is collected with these sensor networks. The portal of the Réseau RUMEUR, shown in figure 4.5, even allows measurements to be tracked in (quasi) real-time.

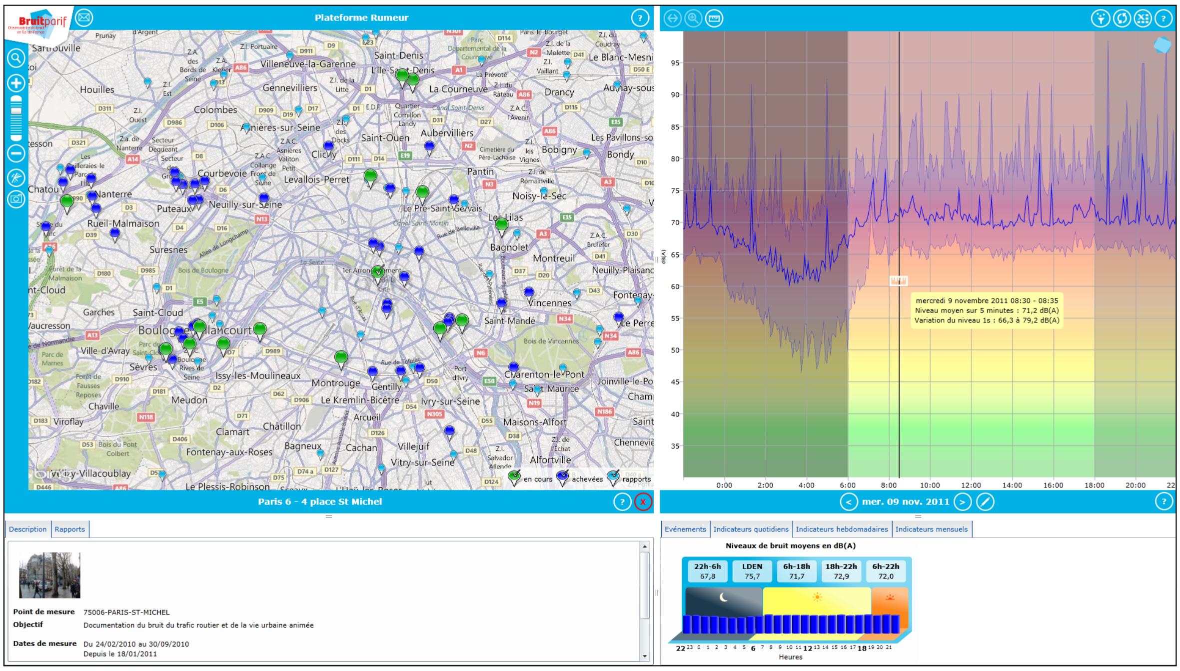


Figure 4.5: Web portal of Réseau RUMEUR, the official noise sensor network in and around Paris, France [59]

Advantages

The main advantage of using sensor networks is that it allows to monitor the actual sound level at certain places accurately over arbitrary long periods of time, which is not feasible with manually operated equipment. The data that is collected with these networks allows officials[22] to study temporal variations in the soundscape of specific places – for instance to discover (or confirm) the existence of patterns recurring on a daily, weekly, monthly or seasonal basis – which is not possible with strategic (simulation-based) noise maps because they lack a local, short-term perspective. As noted above, sensor network data can also be used to validate the output of simulation models.

[22] As well as academics or even citizens, provided that they can access the data.

Limitations

Network sparsity

Considering the size and population of the cities in question, currently operational sensor networks are rather sparse. For instance, the Brussels Region covers over 160 km^2 and is home to over 1 million people, yet the BIM/IBGE network has only 17 nodes, which is not nearly enough to, for instance, compare all neighbourhoods. If installing additional nodes is too costly (see below), making them nomadic allows spatial coverage to be extended, yet at the expense of reduced temporal coverage.

High cost

Due to the high cost of individual nodes deploying a permanent noise sensor network is currently not within reach of NGOs or authorities lacking large budgetary means (e.g. in developing countries). For the same reason it may be infeasible to increase spatio-temporal granularity of existing networks (i.e. make them denser).

Source identification

As we discussed in sections 4.2 and 4.3.2.4, not all sources of noise are equally annoying or harmful. Although officials probably already have an idea of the dominant source for most places in a city, it could still be interesting to verify such assumptions and to know which other sources are present. However, without human intervention (i.e. either by listening in-situ or to recordings), reliable identification of (mixed) sound sources remains difficult and, as far as we know, none of the currently deployed sensor networks can do this. Ironically this is not a problem for simulation models because there each source is modelled separately.

Places vs. people

As discussed in section 3.2.2 (wireless) sensor networks have a location-centric perspective, rather than a people-centric one. This means that sensors, which are usually placed at fixed locations for a least a few days or weeks, are inherently measuring pollution at given places, rather than the exposure of (individual) people during their daily lives and mobility. Moreover, current networks do not provide a way to assess the subjective perception of people living in the neighbourhood.

We should note that in recent years researchers have made efforts to develop technologies that could enable cheaper (and thus potentially denser) noise sensor networks. For instance, Santini et al. have investigated the potential of embedded hardware platforms as used in WSNs [180, 458–460], and Van Renterghem et al. showed that it is also feasible to build nodes with PC hardware and microphones found in consumer electronics [551]. Other interesting research was conducted by Defréville et al. In [118] they describe an algorithmic solution – based on the *EDS* system [612] developed at Sony CSL Paris – for automatic recognition of up to 6 urban sound sources. Building on that work they developed an experimental sensor system called *ORUS* which, besides measuring the ambient sound level, uses identified sources to calculate an unpleasantness indicator [117, 119]. While these are important advances, as far as we know they have yet to be applied in operational sensor networks.

4.3.2.8.3 Community science campaigns

In some places, members of local communities, typically assisted by professional scientists, have set up their own noise measuring campaigns. Such initiatives typically focus on specific local issues (e.g. a factory that is considered too loud) and are organised in parallel with, or in absence of, efforts by authorities. To collect evidence about annoying or harmful sources of noise citizens use sound level meters (SLMs) to measure the ambient sound level at those times and places they consider to be of interest. Simple noise maps can then be made with GeoWeb applications (see section 2.4.3.3.2).

Two successful examples took place in London neighbourhoods affected by airport and industrial activities[23]. The campaigns were coordinated by researchers from University College London and a charity organisation called London 21, which also provided participants with coaching and SLMs. Apart from measuring the sound level participants were also asked to fill out questionnaire forms that asked for qualitative descriptions of the measured sound and their subjective perception of it. Both campaigns were able to bring local noise pollution problems to the attention of policymakers [151, 193, 324, 325].

Advantages

The main advantage of this method is that it allows citizens to collect evidence on perceived problems without having to wait for officials. Moreover, they allow to simultaneously collect quantitative and qualitative data which, as we argued in section 4.2.4, can increase insight into the true impact environmental noise has on citizens' quality of life.

Limitations

Cost/availability of instruments

The fact that sound level measurements must be made with SLMs is a major obstacle for this method. Because these are dedicated, single-purpose devices which usually cost at least €150 (see section A.5.1.1) most citizens are unlikely to be willing (and able) to buy them with their own money. Therefore such campaigns typically rely on funding provided by public or academic institutions or NGOs. Limited budgets may mean that only few SLMs can be bought, which restricts the number of participants and could in turn lead to a biased representations.

[23] We already mentioned these in our discussion of the practice of community science in section 2.4.3.3.3.

Credibility

Average citizens are not trained acousticians. Without the support of professionals, campaigners may have a hard time to analyse the data correctly in order to provide scientific evidence to back their claims.

Lack of continuity and infrastructure

Such campaigns are usually short-term efforts aimed at raising awareness about a local issue. Therefore they lack the continuity required for a long-term vision and sustainable management of the situation. Moreover these initiatives may lack a central (ICT) infrastructure which can act as a repository for historical data and can serve to disseminate this knowledge to the public.

4.3.2.8.4 Fieldwork by officials

So far we have only implicitly referred to possibility of conventional fieldwork by officials. Indeed, environmental agencies (or their subcontractors) sometimes conduct small-scale, short-term acoustic studies at specific places. Similarly to the community science campaigns discussed above, such efforts may be aimed at assessing a local problem, possibly in response to complaints by citizens. Alternatively the goal may be to collect data to initialise or validate modelling efforts. Fieldwork and analysis are carried out by professionals using specialised equipment (i.e. highly accurate SLMs) and software. The results are stored in databases managed by the agency and can be used to advice policymakers. Hence credibility and continuity are ensured. However, this approach does not scale. Authorities simply do not have the means to let their personnel carry out measurements everywhere and all the time, which is why the simulation model and sensor network methods have been developed in the first place.

4.3.2.8.5 Summary

The methods we have discussed all have their own advantages and limitations and are complimentary in many ways. Yet overall there remains much to be desired.

Simulated strategic noise maps provide a general overview but lack spatio-temporal detail and are not updated frequently enough to capture the real experience of citizens. Sensor networks, on the other hand, do allow to study temporal variations but are too costly to be widely and densely deployed. Fieldwork by officials can provide a lot of accurate data on local situations but is too labour intensive to be scaled up in time and space. In general, the methods applied by authorities have little attention for the contextual and subjective factors that influence the perception of noise by citizens.

Community science campaigns have great potential because they involve citizens directly. This makes it easier to raise awareness and to focus on the times and places which are considered to be most problematic. Moreover the simultaneous collection of quantitative and qualitative data can provide additional insights in citizen's perception of the problem. These efforts are a "low-tech" blueprint of what we hope to achieve with community memories, mobile sensing and social tagging. As we announced in sections 2.5.1 and 3.5.1 and will concretise in chapter 5, these technologies and practices have the potential to lower the cost of monitoring by citizens and to enable the establishment of infrastructures to store, analyse, visualise and disseminate data. This could largely compensate the limitations of current community science campaigns and allow them to be scaled up such that they become truly complementary, or a viable alternative, to efforts of authorities.

4.4 Conclusion

In this chapter we have provided a thorough discussion of the phenomenon of noise, covering both subjective and objective facets.

By looking at definitions of noise and drawing upon findings from (psycho-)acoustic research, we learned that whether individual sounds are considered to be noise – and thereby as unwanted, interfering, disturbing or annoying – can depend on a multitude of non-acoustical, subjective factors related to context, semantics, mood, culture and taste. Consequently sound level measurements only provide a narrow, incomplete view on the human perception and interpretation of sounds and soundscapes. Therefore we consider it important that sound level measurements are somehow (e.g. by means of social tagging) complemented with qualitative assessments of soundscapes.

Drawing upon scholarly literature and official reports, including very recent publications by the WHO and EU institutions, we showed that environmental noise is a growing problem in urban and industrialised areas all over the world and has far-reaching social and economic consequences. For instance, the WHO estimates that exposure to environmental noise is causing up to 1.6 million healthy years to be lost in Europe every year. Hence at a societal level noise is a hazardous pollutant and not just a debatable nuisance. Still, more data is required to increase insight in the adverse consequences of environmental noise and to establish policies to reduce it. We also discussed the current response of policymakers as well as currently used assessment methods and their limitations.

Together this constitutes a comprehensive introduction to the themes of noise perception and noise pollution. Along with appendix A this chapter summarises the domain knowledge that enables us to propose a novel, participatory solution to the assessment of environmental noise. This approach is built around the *NoiseTube* system, which we introduce in the next chapter.

Further reading

A multitude of insights about the concept of noise, its meaning, its subjective and objective aspects, its societal and political impacts, etc., can be found in *The Unwanted Sound of Everything We Want* [297] by American writer Garret Keizer.

In *Mechanical Sound: Technology, Culture, and Public Problems of Noise in the Twentieth Century* [51], Dutch historian Karin Bijsterveld sketches the evolution of (environmental) noise and perceptions thereof, throughout the previous century.

In *Making Noise: From Babel to the Big Bang and Beyond* [465], American historian Hillel Schwartz provides a historical, cultural and social perspective on the noise phenomenon.

The forthcoming *Noise Mapping in the EU* [320] (due to be released in August 2012), by Gaetano Licitra of the Environmental Protection Agency of Tuscany (Italy), promises to be a comprehensive reference guide regarding (conventional) noise mapping procedures.

Chapter 5

The NoiseTube system

5.1 Introduction

Now the context of our research has been set, our general vision for sustainable governance of commons facilitated by community memories, mobile sensing and social tagging has been explained, the state of the art of mobile sensing research and applications has been discussed, and the problem of environmental noise has been introduced, it is time to present a concrete solution.

This chapter introduces the NoiseTube system and the participatory approach to the assessment of environmental noise it enables. The NoiseTube system consists of two principal components: the *NoiseTube Community Memory*, which is discussed here, and the *NoiseTube Mobile* app for smartphones, which is treated in detail in the next chapter. In this chapter we also discuss how the system has been deployed and used so far.

First section 5.2 situates the NoiseTube project in terms of timing, involved people, institutes and sources of funding. Next section 5.3 introduces the general approach and compares it with conventional methods for environmental noise assessment. Then section 5.4 introduces the architecture of the NoiseTube system. After that section 5.5 treats the functionality and implementation of the NoiseTube Community Memory. Then in section 5.6 we discuss how, where and by whom the system has been used so far. In section 5.7 we discuss, among other things, our experience with the early deployment of the system. Finally section 5.8 concludes the chapter.

5.2 Situating the project

The NoiseTube system was developed in the scope of a research project, which we will refer to as the *NoiseTube project*, that started in June 2008 at the Sony Computer Science Laboratory (CSL) in Paris, under impulse of the lab director Prof. Dr. Luc Steels. The initial project team consisted of Nicolas Maisonneuve, Maria E. Niessen[1] and myself. Dr. Peter Hanappe, a member of the Sony CSL research staff, took on an advising role.

The project was inspired by the work on community memories and social tagging by Steels & Tisselli [493–495] (see section 2.5) and the initial goal was the rapid development of a prototype for the technologies Sony CSL would have developed in the scope of the *REEACT* project, which was under review at the time. REEACT, short for *Raising European Environmental Awareness through Communal Technologies*, was a proposal for a European FP7 project, drawn up by a consortium of research institutes including Sony CSL Paris. Although the proposal was unfortunately turned down[2], it has nevertheless been a continued source of inspiration for our research.

The abovementioned NoiseTube Prototype, which is discussed in appendix C, was completed in late August 2008. Work on the current NoiseTube system, which is discussed in the following sections and the next chapter, started in September 2008. A first version was released to the public in May 2009. Also in spring 2009, the NoiseTube system was first presented to academic audiences at a workshop [335] and two conferences [331, 332]. In 2010 it was covered in a journal paper [334][3].

The project continued at Sony CSL over the course of 1.5 years during which Maisonneuve coordinated. During this period I spent a total of 9 months as an intern at the lab and I contributed remotely during the other 9 months. In the second half of 2009, Maisonneuve and myself were joined by Bartek Ochab. At Sony CSL, work on NoiseTube was partially funded through *TAGora* [522], a European FP6 project focused on the study of semiotic dynamics in online communities with social tagging systems (see section 2.4.3.1.2).

As of 2010 the NoiseTube project has been coordinated by the newly formed BrusSense team at the Computer Science department of Vrije Universiteit Brussel [66]. BrusSense consists of Dr. Ellie D'Hondt and myself and works under promotorship of Prof. Dr. Steels. Dr. D'Hondt is funded by a grant [124] from Innoviris, the Brussels Institute for Research and Innovation. My own source of funding was an "aspirant" grant from Research Foundation Flanders (FWO–Vlaanderen) which ran from October 2007 till September 2011.

[1] Niessen left after her internship ended in August 2008.

[2] Later, an altered consortium, without Sony CSL Paris, successfully obtained EU funding for a project derived from the REEACT proposal. This project, called *EveryAware*, got underway in March 2011 [175].

[3] Refer to appendix G for an overview of all scholarly and popular publications, media and events in/at which our work has been presented or mentioned.

Within the BrusSense team we have continued the development and maintenance of the NoiseTube system and have taken several new research steps. Most notably we focused on the validation, by means of laboratory and field experiments, of the system itself and the novel, participatory noise mapping method it enables, as discussed in chapter 7. Moreover, we have embarked on a number of related on-going efforts listed in chapter 8. A total of 7 bachelor and master students from the VUB's and ULB's computer science programs have made direct or indirect contributions to our research in the scope of assignments and thesis projects advised by Dr. D'Hondt and myself.

5.3 The NoiseTube approach

The conceptual and architectural requirements for the NoiseTube system follow directly from those discussed in section 3.5.1, but must be concretised in order to design a system that is aimed at the application domain of monitoring environmental noise and people's perception thereof. Concretely we need a mobile sensing system that allows people to measure the ambient sound level using their mobile phone and to simultaneously comment on it by means of social tagging. The system should allow people to share the (geo-)tagged data with others through a community memory (CM). This CM should support the aggregation of data coming from multiple contributors and should facilitate its exploration, visualisation (by means of charts and maps) and dissemination, with the overall goal of supporting the community of users in the construction of a shared, annotated (i.e. tagged) representation of the soundscape in their neighbourhood or city.

In terms of the vision discussed in section 2.5, that soundscape is the commons which is at stake. More generally the commons can be seen as the community's quality of life, which may be under threat from environmental noise as well as other pollutants and nuisances. To some extent virtually all citizens are at least indirectly responsible for some of the noise produced in their neighbourhood (i.e. by shouting, playing loud music, driving a car or riding a bus). Rather than to think of noise as "the sound of others", it is beneficial to reason about it as "the sound of all of us" (see section 4.2). Only then it may be possible to start changing our collective behaviour and solve part of the problem. Therefore, starting from the level of individual participants, the NoiseTube system should support the raising of awareness about noise and its diverse sources.

As explained in section 3.5.1, when the goal is to raise awareness about local conditions it is best to opt for a participatory, rather than an opportunistic, sensing system. Such a system puts individual users in charge of where and when data is collected and provides immediate (on-screen) feedback, in this case about the measured sound level. This direct feedback also invites people to engage in social tagging. When they see measurements in real-time users are more likely to feel the need to "explain" what is happing by adding qualitative descriptions as tags. For instance, a user may want to identify dominant sources of noise (e.g. "truck"), describe context ("@home", "sleeping"), and document his/her perception ("loud", "annoying"). Rather than only serving as a channel for complaints about loud or annoying noises, social tagging can convey the full spectrum of emotions and opinions related to sound perception, including positive experiences (i.e. what is *good* about the soundscape at a certain place), which helps to paint a richer, more balanced picture. In the CM the combination of such qualitative descriptions and quantitative sound level measurements allows to make the latter easier to interpret, search through and explore, and to make noise maps that are more informative than conventional (END-type) ones.

As argued in section 3.5.1, NoiseTube should support usage at different, nested levels of scale: individuals as well as groups should be able to use it, whether or not they contribute to larger, mass sensing efforts (e.g. the creation of city-wide noise maps). To reduce privacy concerns, users must stay in charge of their data, even at the level of the CM. When a user uploads data it should not necessarily be shared with others and users should be able to consult (and possibly remove) the history of their own contributions.

In comparison with conventional methods for the assessment of environmental noise, discussed in section 4.3.2.8, a number of differences stand out:

Democratisation of environmental monitoring

When they are confronted with loud or annoying sounds citizens typically have no way to know "how loud" those are, let alone what their average daily/nightly exposure is. In section 4.3.2.8 we discussed community science initiatives which try to answer such questions by making measurements in the streets with sound level meters. However, the high cost and complexity of these devices is a limiting factor for such initiatives. By turning the mobile phone – a relatively cheap and simple, off-the-shelf device which most people already own – into a personal sound level meter, we strongly lower the barrier to the assessment of environmental noise.

Measuring vs. simulating

To produce of city-wide noise maps authorities until now have had to rely on (possibly outdated) statistics for limited numbers of sources. Provided that enough people can be convinced to participate in densely-populated areas, measuring campaigns using NoiseTube have the potential to collect enough fine-grained data such that similar maps can be made using recent measurements covering all sources.

People-centric perspective

As already discussed in section 3.2.2, mobile sensing systems allow to monitor the environment from a people-centric perspective. Because mobile phones are virtually always co-located with an individual person, what is measured is not just the sound level at certain times and places but also the exposure of the person in question.

This allows the system to provide personalised feedback, which can have a much bigger impact on people's awareness than general statistics provided by authorities.

As explained in section 2.5.1, initiatives to set up a CM and organise participatory data collection campaigns can come from both citizens or authorities.

The scale of citizen-led initiatives can range from individuals who use NoiseTube on their own, to groups of fellow citizens with varying degrees of cohesion and coordination: from total strangers that happen to live in the same area, over loosely organised groups of neighbours, to well-organised existing activism groups. Motivations can be diverse: from curiosity about one's daily environment to the gathering of evidence on concrete local issues. These can be long-term issues (e.g. the problems faced by people living close to airports, highways, factories or nightclubs); short-term ones (e.g. roadworks or construction sites); or incidental annoyances (e.g. demonstrations). In absence of environmental noise assessment efforts by authorities[4], participatory noise mapping using NoiseTube can provide citizens with a viable alternative. There where authorities use conventional assessment methods, citizens' efforts can be seen as complementary due to the different perspective (i.e. people- vs. location-centric) and the ability to collect data in places that are not covered or accessible by official initiatives (e.g. in private homes). Whether alternative or complementary, participatory noise mapping empowers citizens by allowing them to collect fine-grained data to convince authorities, not only to establish policies to reduce noise exposure but also to protect those soundscapes that are considered pleasant.

Authorities, typically on a municipal or regional level, can take the initiative to organise campaigns in which citizens[5] use NoiseTube, or similar/derived solutions, to collect quantitative and qualitative data about environmental noise. The required number of participants – and the time they need to invest – depends on the desired spatio-temporal scale and granularity. While authorities can choose to work exclusively with (noise-concerned) volunteers, it may take significant publicity, communication and coaching efforts to attract enough people and to keep them motivated, active and compliant[6]. Therefore it may be necessary to consider schemes in which contributors are provided with monetary or other incentives[7]. A possibility could be to offer free calling minutes or SMS messages in return for contributions, or a leasing scheme in which each volunteer is given a smartphone that can become his/her own if he/she remains active during a certain time period. Other schemes could involve publicity deals with advertisers or cellular network operators. Authority-led participatory noise mapping campaigns can complement or be integrated with existing (conventional) assessment efforts. However thanks to the relatively low cost

[4] For example, in smaller European towns which are not (yet) included in the END or national legislation, or in agglomerations in developing countries which generally still lack such legislation.

[5] It is also conceivable that all or some of the fieldwork is carried out by officials, rather than citizens.

[6] For instance, compliance could mean that participants make measurements when and where they are told, rather than only when/where they consider the noise level is problematic.

[7] As mentioned in section 3.4, work by Reddy et al. indicates that such schemes can work [439].

they are also within reach of authorities that currently have no assessment policy in place due to limited budgetary means. When used alongside existing efforts, participatory noise mapping could make up for missing data (i.e. places or times not covered by simulation models, sensor networks or fieldwork by officials), provide *real* data on the daily exposure of citizens *and* their perception of it, add semantics (e.g. source identification), etc. Moreover, such campaigns can make noise management polities more transparent due to the direct involvement of citizens in data collection, interpretation and dissemination.

To summarise, table 5.1 provides a comparison of the NoiseTube approach with the conventional methods discussed in section 4.3.2.8:

	Simulation models	Sensor networks	Fieldwork by officials	Community science campaigns	NoiseTube
Inputs	Statistics concerning limited number of noise sources + information on urban topology	Sound level measurements made by a limited number of sparsely distributed highly accurate autonomous SLMs	Sound level measurements made at specific places during short periods, by officials w/ professional-grade SLMs	Sound level measurements collected w/ SLMs in specific area during short periods + qualitative info.	Sound level measurements made w/ mobile phones (less accurate than SLM) + qualitative info. + GPS coordinates
Operators	**Authorities, public institutions, environmental agencies**			**NGOs, small citizen groups**	**Potentially everybody** Initiative can be citizen- or authority-led
Output	Strategic noise maps, each of which usually represents a single specific sound source	Daily or hourly indicators, sometimes real-time measurements	Reports and maps about local issues		Measurements enriched through social and automatic tagging, different types of aggregated maps, noise exposure profile for individual users, …
Perspective	**Strategic** Captures long-term trends, not local/short-term variations	**Location-centric, long-term** Allows to continuously monitor temporal variations at a limited number of places	**Location-centric, short-term** Allows to investigate local situations over short periods of time	**People-centric, short- or long-term** Measuring equipment is co-located with citizens going about their daily routines, participants collect as much data as they want or have time for, dense spatio-temporal coverage if enough people take part, coverage may be limited by their mobility patterns (but noise at places no-one visits is less problematic anyway)	
Deployment cost	**High** Requires access to large amounts of data, simulation software, powerful computers, expertise	**High** Requires expensive sensors and sometimes a (wired) communication infrastructure	**High** Requires expensive professional-grade SLMs and trained personnel	**High/Moderate** Requires costly SLMs and coaching for participants (this usually restricts participation to a small group)	**Low** Uses relatively cheap, off-the-shelf devices (i.e. mobile phones) which many people already own
Dissemination	Maps are usually published online (no assess to raw data)	Indicators published online (usually no access to raw data)	Reports may be published online (usually no access to raw data)		Community Memory website, semantic exploration, down-loadable maps, access to raw data, feeds, e-mail alerts, …

Table 5.1: Summarising comparison of the NoiseTube approach with conventional methods for assessment of environmental noise

As discussed in section 2.5.1, academic researchers can play several roles in the development of CM systems, their deployment and the organisation of associated data collection campaigns for/with citizens and/or authorities. Besides developing the NoiseTube system we have also played an active role in its deployment, as we discuss in section 5.7.

5.4 Architecture

The architecture of the NoiseTube system, illustrated by figure 5.1, is similar to the prototypical mobile sensing architecture discussed in section 3.3.1. Hence it follows the typical client-server model, in which the client-side is formed by an application installed on mobile phones[8], and the server-side is formed by dedicated software running on a machine that is permanently connected to the Internet. Our client application is called *NoiseTube Mobile* and the server software is called the *NoiseTube Community Memory*.

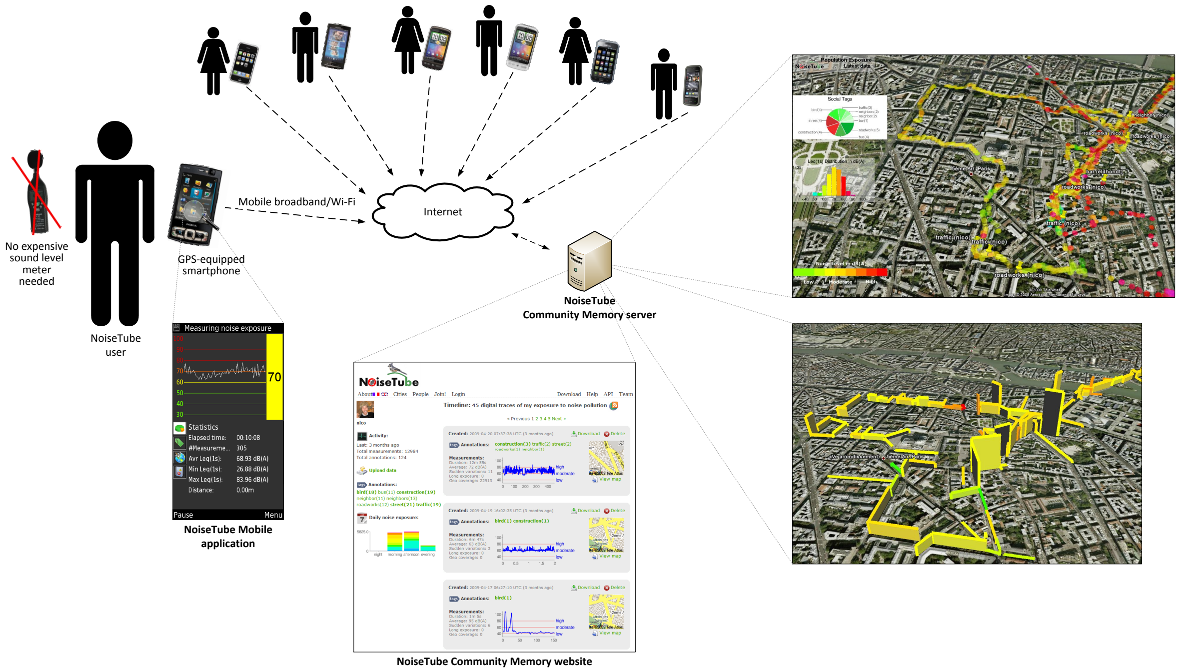


Figure 5.1: Architecture of the NoiseTube system

By installing NoiseTube Mobile on their smartphone people effectively turn it into a personal sound level meter. It allows people to make *tracks*, which are series of time-stamped, geo-tagged[9] sound level measurements, to which they can add social tags and which they can submit to the (or a) NoiseTube Community Memory. Submitting data is optional, which means that users can just as well use NoiseTube Mobile as a stand-alone, personal sensing app. Clients can currently not interact or communicate amongst themselves (i.e. in a peer-to-peer fashion)[10]. NoiseTube Mobile is available for multiple platforms (Java ME, Android and soon iOS) and is discussed in detail in chapter 6.

On the server-side the *NoiseTube Community Memory* (CM) is a Web-based community memory system aimed at the assessment of environmental noise. It acts as a central

[8] Also on the client-side are people who consult the NoiseTube Community Memory using a Web browser.

[9] I.e. associated with geographical coordinates obtained from a GPS receiver.

[10] However this possibility was explored in the scope of a master's thesis [39] (see chapter 8).

repository for data collected by users of NoiseTube Mobile. A thorough discussion of the functionality and technical aspects of the NoiseTube CM follows in the next section. The communication protocol (and associated file formats) for data exchange between NoiseTube Mobile and the NoiseTube CM is documented in appendix D.

Currently there is just one running instance of the NoiseTube CM. It is reachable at http://noisetube.net, which doubles as the website about the NoiseTube project [375], and has been online and open for use by the public since May 2009. Until July 2010 it was hosted on a server at the offices of Sony CSL Paris, now it is hosted at the VUB and managed by the BrusSense team. As we discuss in sections 5.6 and 5.7 this service is being used by people living in different places around the world. However, because the problem of environmental noise is typically assessed at a city level and to support local collaboration, the NoiseTube CM aggregates shared data per city.

In June 2011, upon receiving formal approval of Sony France[11], we made NoiseTube an open source project. The source code of the NoiseTube Community Memory – and that of NoiseTube Mobile – was published on Google Code[12] [378] under the terms of the LGPL v2.1 license [195]. This makes it possible for others to set up their own separate instance of the NoiseTube CM – with or without our help – for example to use it for a local noise mapping campaign. While, as far as we know this has not happened yet, it is important for at least two reasons. On the one hand the emergence of multiple NoiseTube CMs would solve the current scalability and stability risks associated with having a single point of failure. On the other it would solve or reduce a number of concerns related to privacy – i.e. some people may be more conformable with sharing data through a service that is operated by a local organisation that they know and trust. More considerations regarding scalability, privacy and the advantages and disadvantages of having one or multiple NoiseTube CM instances are discussed in section 5.7.

5.5 The NoiseTube Community Memory

Here we discuss the main functionality and the important technical aspects of the implementation of the NoiseTube Community Memory software.

5.5.1 Functionality

NoiseTube Community Memory handles the reception, processing and aggregation of data submitted by users of NoiseTube Mobile. Moreover it serves as a Web portal [375]

[11] Acting as the legal parent company of Sony CSL Paris.

[12] A hosting service for open source software projects.

that offers tools to explore, visualise, analyse, search through and disseminate results. The portal also lets users create a personal account (necessary to submit data), download the NoiseTube Mobile app, manage their contributions and edit their account details.

5.5.1.1 Processing

Once tracks are submitted to the CM – which can happen in multiple ways, as discussed in chapter 6 and appendix D – the data is processed in several steps:

Basic processing

First the track goes through a number of basic processing steps. For instance: the number of measurements is counted and tracks with less than 3 measurements are discarded (too short); the track's duration is computed based on the timestamps of first and last measurement; measurements with unrealistic sound levels are removed[13]; the minimum, maximum and average[14] sound level are computed; a chart image showing the sound level variation is generated. If the track has at least 2 measurements which are associated with GPS coordinates we also compute the track's total distance and a spatial bounding box (minimal and maximal latitude and longitude). Next, we determine in which city or commune the track was made. To do this we query a reverse geo-coding[15] service provided by Google [217] with the coordinates of the first measurement that has them, which gives us a street address including the city name. If the city is already known in the system the track is associated with it; if not, we first add a new city record to the database. If there are no geo-tagged measurements we associate the track with the city the user specified upon creating his/her account[16].

Automatic contextual tagging

As explained above, NoiseTube Mobile allows people to tag measurements, for instance to identify sources of noise, but also to describe the context in which the data was collected. However because entering tags takes time and may be impractical in certain situations, people may not do it frequently enough to provide sufficient information to allow others to understand the context in which measurements were made. Fortunately, certain contextual parameters, such as those

[13] The current minimum is 20 dB(A) and the maximum is 130 dB(A), but these could be changed easily.

[14] As the overall L_{Aeq} for the duration of the entire track, see section A.5.1.3.2.

[15] Geocoding is the process of finding geographic coordinates from other data, such as a street address. Reverse geocoding is the opposite: finding an associated textual location (typically, but not necessarily, a street address), from geographic coordinates [583].

[16] When users collect data outside their hometown, and the optional geo-tagging by GPS is disabled (see chapter 6), then the association is wrong. However, in absence of geographic coordinates this cannot be detected automatically. In the future we may add a feature that allows users to manually associate such tracks with the correct city. Although in a sense this is already possible by means of tagging.

related to time and location, can be inferred automatically, mainly based on timestamps and GPS coordinates. Hence we have implemented a processing feature that automatically generates contextual tags for all incoming data. In this manner the machine complements the human tagging effort.

The automatic tagging feature uses a set of classifiers to generate additional tags describing different contextual dimensions such as time, location, user mobility, weather conditions, etc. For instance, the location-related classifiers take GPS coordinates as input and use the abovementioned reverse geo-coding service [217] to generate tags such as postal codes, city and street names. Another example are the time-related classifiers which generate tags to indicate the moment of the day (e.g. "afternoon"), day of the week ("Monday"), type of day ("weekday", "weekend"), the season, etc. Another classifier uses the combination of timestamps and GPS coordinates to calculate the travel speed, from which it guesses if and how the user is moving ("stationary", "walking", "using transport"). Yet another classifier uses time and location to determine local weather conditions by querying a weather report service. Weather information is useful for two reasons. First, unlike sound level meters mobile phones do not have a windscreen covering the microphone, hence measurements made in windy conditions can overestimate the actual ambient sound level. By storing weather information as tags it becomes possible to filter out measurements made in windy conditions or give them a lower weight in aggregations. Second, as we mentioned in section 4.2.3, researchers have found that the weather can influence people's perception of urban soundscapes [611]. Below we show how manually and automatically generated tags are applied to aid users in navigating and searching through the data.

GPS correction (experimental)

Typically the coordinates determined by a GPS receiver have an error margin of anywhere between 5 and 20 meters. Especially in dense urban areas GPS errors can become problematic due to the so-called canyon effect [252]. To make up partially for these inaccuracies the NoiseTube CM has an experimental processing feature that corrects GPS coordinates using a technique called *map matching*. Relying on a GIS database of street segments the algorithm basically "pulls" any point that does not lie on a street segment to the nearest position that does – unless it is too far away from any known segment. Figure 5.2 illustrates the process.

This means we assume that all outdoor measurement are indeed made in streets. While this is often the case, it certainly is not always. Hence, for measurements that are *not* made in streets (e.g. on rooftops, in gardens, parks or fields, etc.) the GPS correction feature is potentially damaging. For this reason it is currently disabled. If we re-enable it in the future, we will probably make it so that the user who submitted the track has to explicitly request for the correction to be applied, rather than applying it automatically for all incoming tracks. We should also note that

(a) Before correction **(b)** After correction

Figure 5.2: Correction of GPS coordinates through map matching

this feature was prototyped using a database of streets in Paris, which we obtained through the city council. Hence it only worked for data collected in Paris. In order to work elsewhere we would need digital street maps of those places. Thanks to free services like OpenStreetMap [399] this is now feasible. However we have yet to implement a mechanism that queries or downloads data from OpenStreetMap.

5.5.1.2 Elog & profile pages

After all processing steps have finished, tracks are added to the submitting user's board of activity, called the *Elog*, short for "Exposure Log" of "Environmental Log". Inspired by the concept of blogs, an Elog collects traces of the environmental dimension of a citizen's daily activity. An example in shown in figure 5.3. After logging in to the Web portal users are automatically taken to their Elog page. Alternatively they can go there by clicking on the "Your Elog" link in the menu shown at the top of the website.

The Elog shows a summary box for each track submitted by the user in question. These boxes show tag clouds[17] of social tags and different categories of automatic tags, the duration and distance of the track, the average sound level, a chart showing the sound level variations, and, if the track is geo-located[18], a small map of the area in which the data was collected and a link to a downloadable map. On such downloadable track maps, of which figure 5.4 shows an example, every sound level measurement is represented by a coloured circle and if the user entered any tags those are also displayed. The map files use the KML format [298] and can be opened in Google Earth [215]. The raw data of each track – consisting of sound level measurements, timestamps, coordinates and tags – can also be downloaded as a file in the JSON format [104].

[17] As explained in section 2.4.3.1.2, a tag cloud is a graphical representation of the frequency of usage of tags: the bigger the size of a tag in the cloud, the more popular it is.

[18] I.e. if it has at least two measurements which are associated with GPS coordinates.

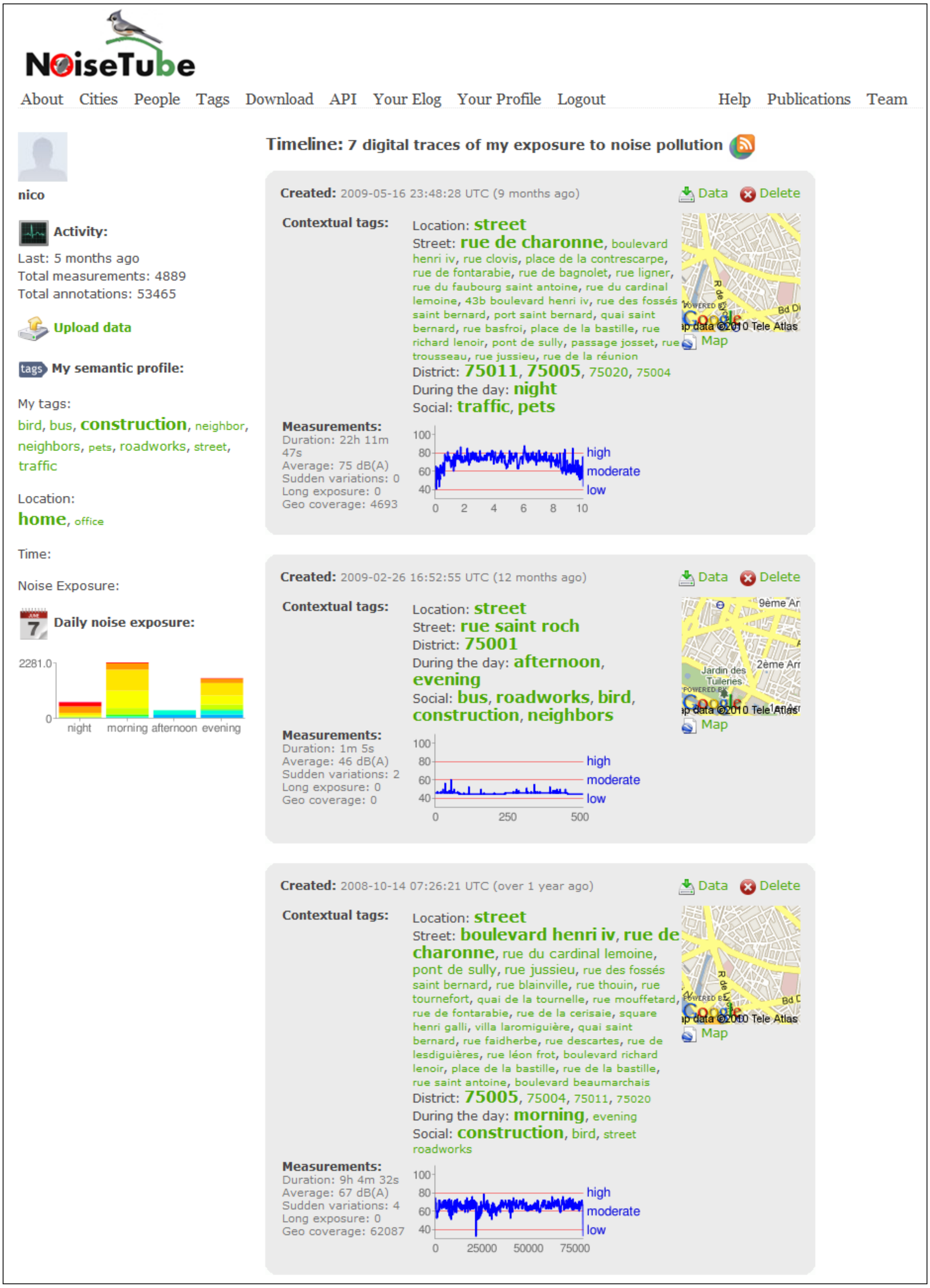


Figure 5.3: Elog page of a NoiseTube Community Memory user, showing 3 track summaries

On the left side of the Elog page we see a summary of the user's overall activity, his/her "semantic profile" (overall tag clouds), and typical daily exposure to noise.

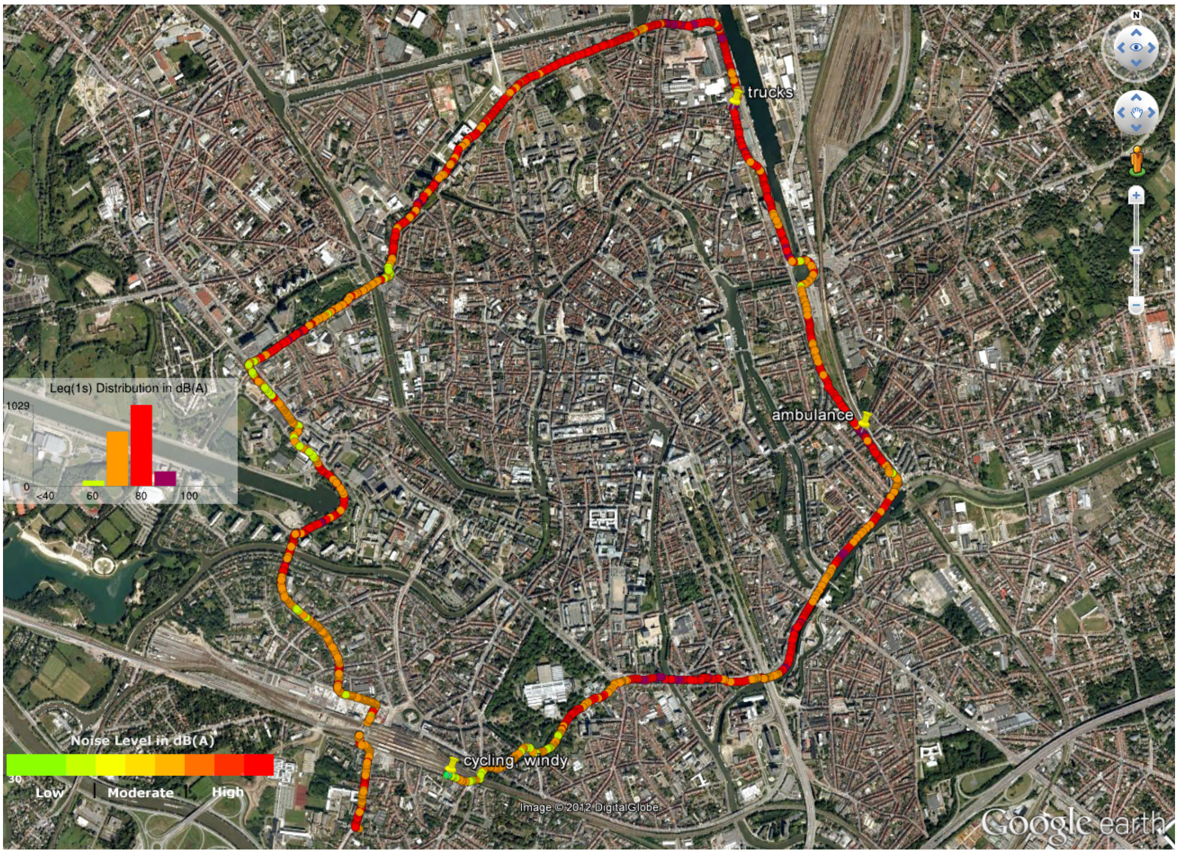


Figure 5.4: Map of a track made in Ghent, Belgium; displayed in Google Earth

By pointing others to their Elog (e.g. http://noisetube.net/users/mstevens) users can show them a representation of their exposure to noise – or sound. However, not all tracks are necessarily visible to others (i.e. shared). On their profile page, which is accessible via the "Your Profile" link (after logging in), users can specify whether they want new tracks to be shared or not. Tracks which are not shared are stored on the CM but are only visible to the user who uploaded them (i.e. the owner). To "un-share" a track the owner can click the corresponding "Delete" link on his/her Elog[19], as shown on figure 5.3, which completely removes the track from the CM database. In the future we could improve this by allowing users to (un-)share tracks without having to delete or re-upload them.

When they click on the "People" link visitors of the Web portal get an overview of all registered users (sorted by recent activity) and links to their Elogs. By browsing through other people's Elogs one can for instance look for peers – i.e. people leading similar lives and/or facing similar problems. However, we have yet to add communication features that would allow users to contact one another through the CM in order to get acquainted, coordinate efforts or debate results online.

[19] Obviously the "Delete" links are not shown when someone else is looking at the Elog page.

5.5.1.3 City-level communities

When they click on the "Cities" link at the top of the website (see figure 5.3) visitors are presented with the overview shown in figure 5.5, which lists the cities (or communes) for which the CM has received data. In a sense, each of these represents a "community within the community". Compared to the community of all users, such city-level communities are closer to how we envisioned a community memory would be used in section 2.5.1 – i.e. by a group of citizens living in the same neighbourhood, village or city.

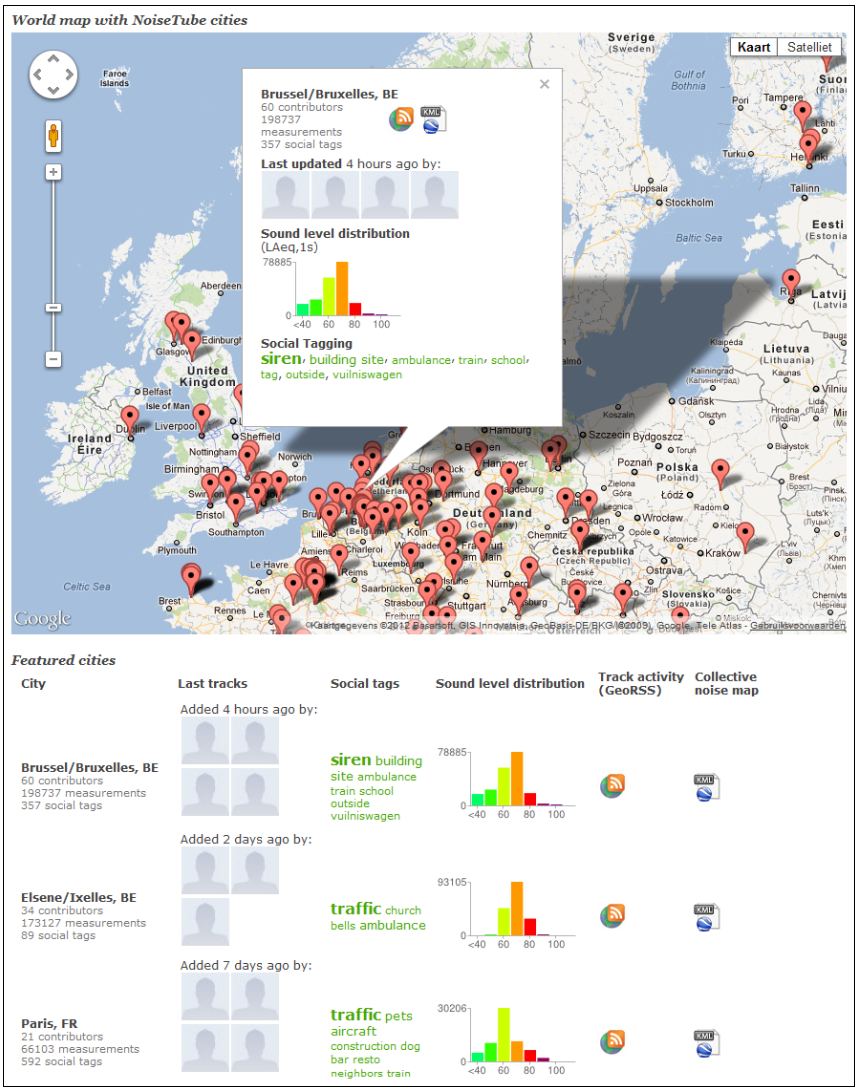


Figure 5.5: Cities overview page in the NoiseTube Community Memory

For each city a list of recent contributors[20], a tag cloud with common social tags, and a chart showing the sound level distribution are shown. Moreover there is a link to a GeoRSS feed [396], which allows visitors to subscribe to updates (i.e. new tracks) for that particular place. Last but not least there is a link to a downloadable "collective noise map", again in the KML format [298]. Figure 5.6 shows an example of such a map.

[20] Clicking on one of the user icons takes the visior to that user's Elog.

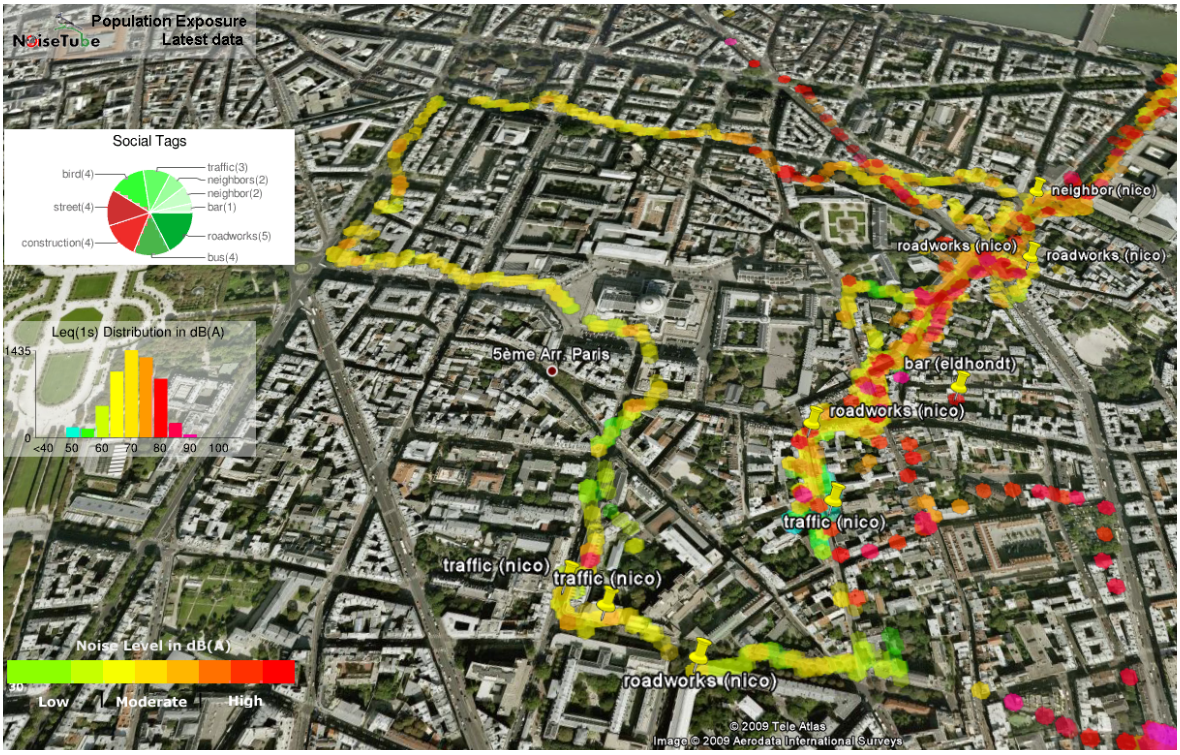
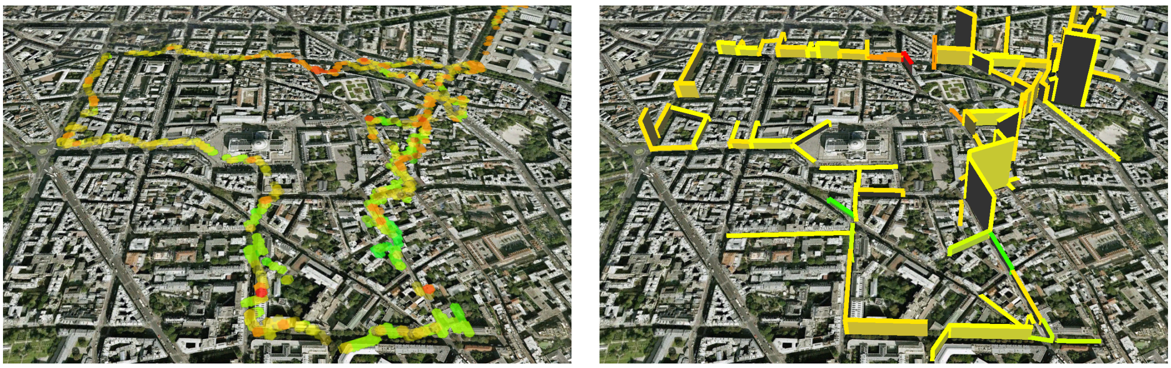


Figure 5.6: Collective noise map for Paris (detail: 5th arrondissement)

The collective noise maps are generated from all shared, geo-located tracks made in the city in question. All sound level measurements are shown as individual coloured circles. To add context and meaning to the numerical data, the maps include a semantic layer (consisting of the social tags) and legends (distribution of the sound level, and distribution of the social tags) that change dynamically according to the area displayed.

The NoiseTube CM can also generate maps in which measurements are aggregated by geographical entities such as street segments or city districts. Figure 5.7 shows an example of aggregation by street segments. The colour of the lines signifies the average sound level measured at each segment. Their height above the ground is proportional to the number of measurements for the segment, and hence gives an indication of credibility.



(a) Before aggregation **(b)** After aggregation

Figure 5.7: Aggregation of sound level measurements by street segment

If there is sufficient data about a given area aggregated maps are much more informative (and easier to interpret) than maps showing individual measurements (which can become too dense). However this feature has the same limitation as the GPS correction discussed above, namely that it requires a digital map of each city. For the same reason it currently only works for Paris. We should note that for the field experiments discussed in chapter 7 we developed a separate, grid-based aggregation tool – which is applicable anywhere.

5.5.1.4 Semantic exploration

Besides the overviews of people and cities the NoiseTube CM offers a third perspective on the data. The innovative "semantic exploration" feature relies on the combination of social tags (entered by users) and automatic tags (generated by the CM) to provide a semantic and contextual perspective. The feature can be accessed by clicking on the "Tags" link at the top of the website (see figure 5.3), or on any tag in the tag clouds shown in the abovementioned other parts of the website. The user is then presented with a number of tag clouds, as shown in figure 5.8. One of the clouds lists social tags, the others contain automatic tags, grouped in different categories (contextual dimensions like location, time, weather) and sub-categories.

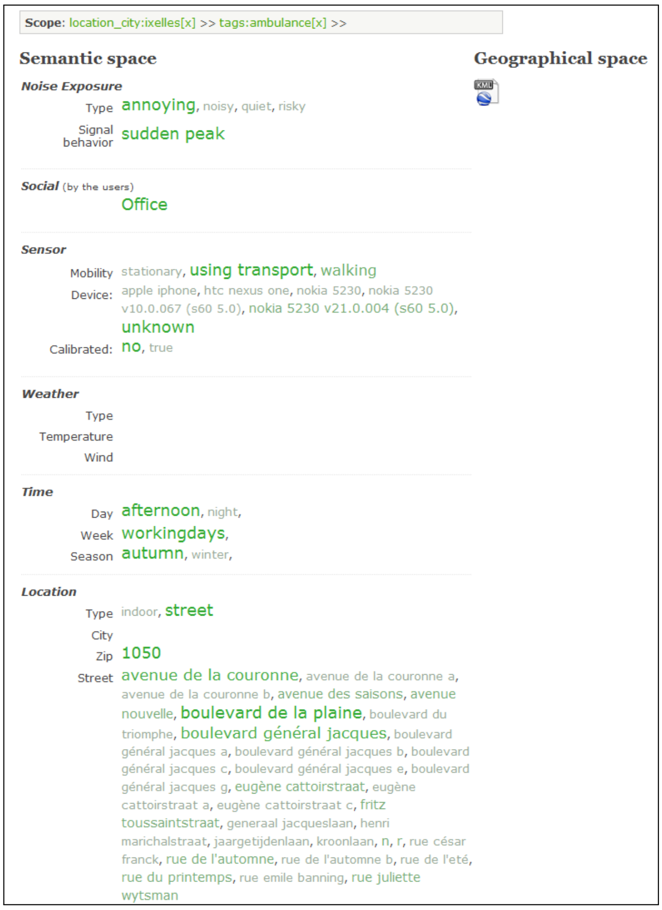
If the user reaches the semantic exploration feature by clicking on the "Tags" link, the initial tag clouds are generated based on *all* data in the CM[21], as shown in figure 5.8a. Conceptually this is similar to looking at a world map from a distance. To focus, or "zoom in", on a specific interest the user can now select tags by clicking on them. Every time a tag is selected the clouds are re-computed based on that subset of the data which is associated with the selected tag(s). For example, after selecting "Ixelles" and "ambulance", one would get the page shown in figure 5.8b, in which the clouds only show tags that co-occur with the selected couple. To "zoom out" users can remove tags from the selection one by one[22]. If the user reaches the semantic exploration feature by clicking on a tag elsewhere on the website that tag will already be selected and the clouds will be based on the subset in question. From there on, he/she can "drill down" to more specific subsets by selecting additional tags, or remove tags to return to a more general level. When at least one tag is selected a link to a dynamically generated map (a KML file) appears, allowing the user to visualise the result of his/her query. Later we may add the possibility to download a JSON file containing the raw data of the selected subset.

This feature allows users to express very specific queries, which are impossible to foresee beforehand, in a simple and intuitive way, resulting in maps that may more meaningful to them than city-level ones. For instance, if someone wants to compare morning traffic noise on weekdays with that during weekends for his/her neighbourhood, that is just a

[21] I.e. all shared tracks, irrespective of who submitted them or where they were made.

[22] Removing a tag is done by clicking on the "X" next to it, as shown at the top in figure 5.8b.

(a) No tags selected



(b) Two tags selected

Figure 5.8: Semantic exploration of the data stored in the NoiseTube CM

matter of selecting the right tags – e.g. "traffic", "morning", "Paris", "75005" and "weekday"/"weekend" would give data tagged with "traffic" collected in the morning in the 5th arrondissement of Paris during the week or the weekend. Moreover, because the queries are specified in the URL[23] users can bookmark them in their browser in order to quickly return to them later on. We should note that, for the moment, queries can only be constructed by means of conjunction – i.e. the AND operator – of tags. In the future we might extend this with support for disjunctions and negations – i.e. OR and NOT.

[23] For example: http://noisetube.net/tags/filter?tags=tags:traffic,location_city:paris

5.5.1.5 Outputs & dissemination

As discussed in section 2.5.1 it is important that a community memory system enables the community to generate and diffuse various types of reports, maps and visualisations, showing different aspects of the situation and/or aimed at different audiences. This has been an important concern during the development of the NoiseTube CM.

We already mentioned that the system can generate different types of maps, either by track, by city or according to a tag-query, and allows people to subscribe themselves to cities by means of GeoRSS feeds. Moreover, it provides access to the raw data of individual tracks, allowing NoiseTube users, or third parties, to generate their own types of maps, visualisations or reports. What we have not mentioned yet is that the system also sends out e-mail alerts[24] and exposes a Web API that enables programmatic access to raw data. This API, which is documented in appendix D, allows third party developers (provided they have a NoiseTube account) to query data by city, submitting user, associated tags, geographical area, minimum or maximum sound level, etc. Results are returned in the JSON format, which is trivial to parse in any programming language.

We have also experimented with social network integration, for instance to allow people to report about their exposure to noise via Twitter or Facebook. These features are currently disabled (mainly due to API changes) but could be brought back fairly easily.

Finally it is also noteworthy that we have developed a number of more interactive interfaces to demonstrate the NoiseTube system at scientific or popular events. This includes a map which shows the position of NoiseTube Mobile users and the sound level they are measuring in real-time, custom city maps on which GPS correction and aggregation of individual measurements to street segments was shown as an animation[25], and a dynamic, self-updating map showing recent measurements by users all over the world[26].

5.5.1.6 Summary

This concludes our overview of the functionality of the NoiseTube Community Memory[27]. While the current system does not include all elements envisioned in section 2.5.1 and there is certainly room for improvements or extensions – e.g. enabling communication among users, or creation of "groups" for focused campaigns – it has the core functionalities a community memory system should have, namely storage, processing, visualisation, exploration and dissemination of citizen-collected and -annotated data.

[24] Users get an e-mail when their tracks are processed, and optionally when others have submitted data collected in their hometown.

[25] Such an animation can be seen in this video: http://youtu.be/GzaOtyjozGs

[26] See http://bxl.noisetube.net/expo (requires the Google Earth [215] browser plug-in to be installed).

[27] A more user-oriented explanation can be found in the NoiseTube User Guide [496].

5.5.2 Implementation

Here we discuss the main aspects of the implementation of the NoiseTube Community Memory, the architecture of which is illustrated by the diagram in figure 5.9.

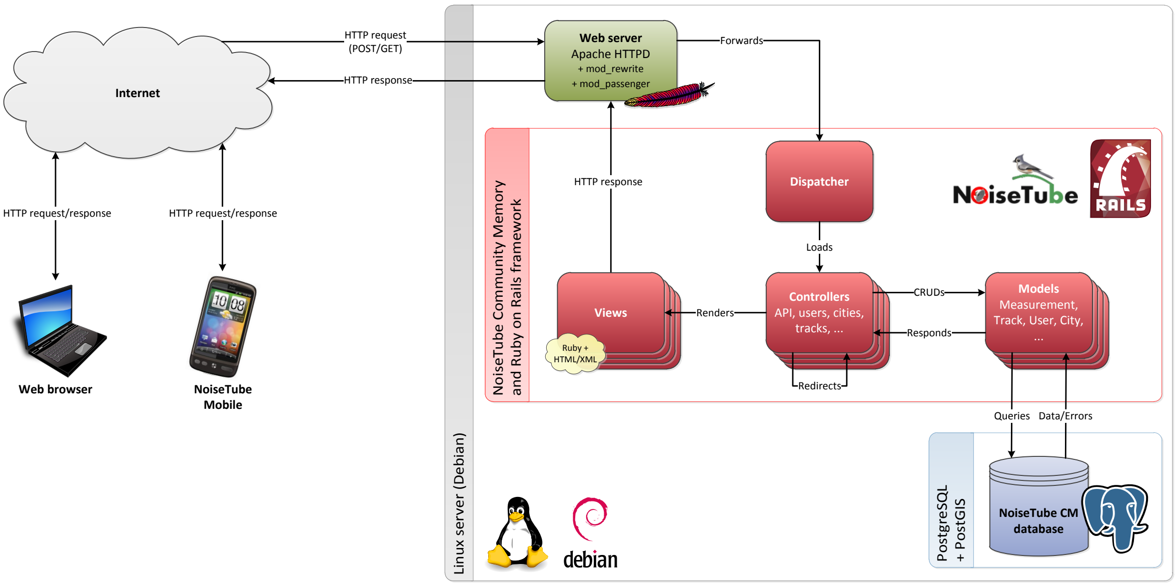


Figure 5.9: Architecture of the NoiseTube Community Memory

Apart from some external services upon which it relies (e.g. the *Google Maps API* [217]) the NoiseTube CM is completely based on open source technologies. The software itself is built using the *Ruby on Rails* [250] framework and is written in a combination of Ruby [337], HTML, CSS and bits of JavaScript/ECMAScript.

Ruby on Rails, often shortened to Rails, is an open source web application framework for the Ruby programming language. We initially chose it because it is well-suited for rapid prototyping – mainly because it is strongly influenced by the *Convention over Configuration* and *Don't Repeat Yourself* principles. This is especially apparent in the way Rails – and the underlying *Active Record* library[28] – lets applications interact with relational databases. If certain rules regarding the names of database tables and columns are respected, Rails infers the database schema and automatically generates corresponding model classes with very little configuration needed. Moreover, it allows programmers to make changes to the data model of an application just once through a system of "migrations". Such a migration is a simple Ruby class in which data model changes (e.g. the addition of a column) are declared. Upon (re)deployment of the application any new migrations are automatically applied to both the database and the application itself. This facilitates the coevolution of applications and their backend databases.

[28] The Active Record library [249] is a Ruby implementation of the *object-relational mapping* (ORM) pattern by the same name as described by Fowler [190: p. 160].

As shown in figure 5.9, HTTP requests, coming from Web browsers (people visiting the website) or NoiseTube Mobile, are not directly received by the Rails application but first pass through the *Apache HTTP Server* [26][29], the de facto open source HTTP server. In principle Apache could easily be replaced by another HTTP server (e.g. *nginx* [520]). In the backend sits a *PostgreSQL* [427] database with the *PostGIS* [447] spatial extension. We chose the PostgreSQL DBMS specifically in order to use PostGIS, which adds data type support for geographic objects (i.e. points, lines and polygons) and allows various types of spatial queries (e.g. distance calculations) to be handled at the SQL level. Currently everything runs on a machine with the *Debian* Linux distribution [113] as operating system. However, with relatively minor adjustments it could run on other operating systems as well (e.g. another Linux flavour, Windows or Mac OS X).

The Rails framework relies heavily on the *Model-View-Controller* (MVC) pattern [446] and hence so does the NoiseTube CM. The main interactions between models, controllers and views are shown in figure 5.9. To handle a request the dispatcher first loads the right controller. For instance, API requests, which start with http://noisetube.net/api/ and mainly originate from NoiseTube Mobile (see appendix D), are handled by the `APIController`, and a request for an Elog page (e.g. http://noisetube.net/users/mstevens) is handled by the `UsersController`. The controller then uses model objects – which represent database records and are dynamically instantiated by Rails/Active Record in response to CRUD[30] queries – to access and possibly modify data, and renders one of the views to construct the response that is sent back to the requestor. The views are written in a mixture of Ruby code and HTML (for Web pages) or XML (for KML maps or GeoRSS feeds)[31].

The entity-relation diagram in figure 5.10 shows the core data model of the NoiseTube CM. Most things are obvious from the functionality described above: each user has a city (his/her hometown) and can be the owner of multiple tracks. Each track was made in a certain city and has a series of measurements. What may be less obvious is the role of the tag, tagging and tagged interval entities. A tag entity represents an actual tag (i.e. a word). A tagging is an association of a tag, a user and a "taggable".

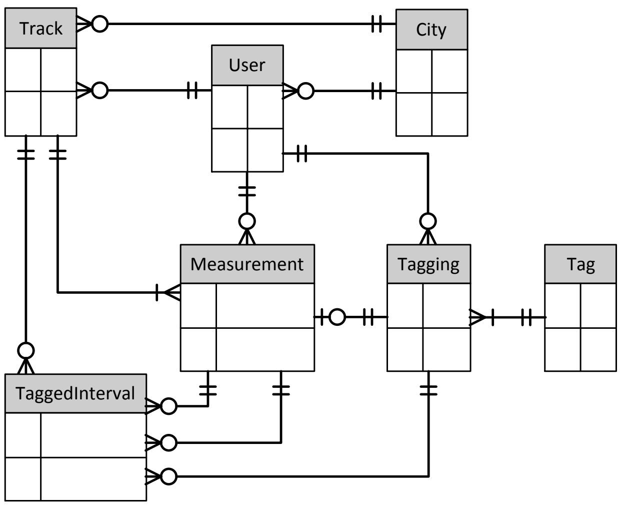


Figure 5.10: Data model of the NoiseTube CM

[29] We make use of 2 additional modules for Apache: *mod_rewrite* [27] to filter out unwanted requests from (search engine) crawlers – only those which trigger queries that cause heavy CPU load – and *Phusion Passenger* (*mod_passenger*) [423] to forward requests to the Rails application.

[30] *Create, Read, Update, Delete.*

[31] API responses are directly generated by `APIController` (in plain text or JSON) without a view class.

A taggable can be either a measurement or a tagged interval. A tagged interval is defined by a track and two measurements (i.e. the first and last of the interval). This entity was added only recently to make the storage of automatic taggings more efficient[32], and to allow users to tag series of measurements instead of one at the time, which is often more intuitive (see chapter 6). Each tagging record also indicates whether it is the result of a user action (i.e. social tagging), and if not, in which (sub)category of automatically generated tags it fits. Currently the relation between tagging and user is redundant since measurements and tagged intervals are already related to a user via their track. However, when the need arises, this would enable us it let users tag each other's data.

5.6 Usage statistics

The one (and only) running instance of the NoiseTube Community Memory [375] was opened for use by the public in May 2009 and at the same time a first version of NoiseTube Mobile was offered for download. Since then almost 1300 people from all over the world have registered an account in order to use, or at least try out, the NoiseTube system. As shown by chart 5.1, the number and rate of registrations continues to increase.

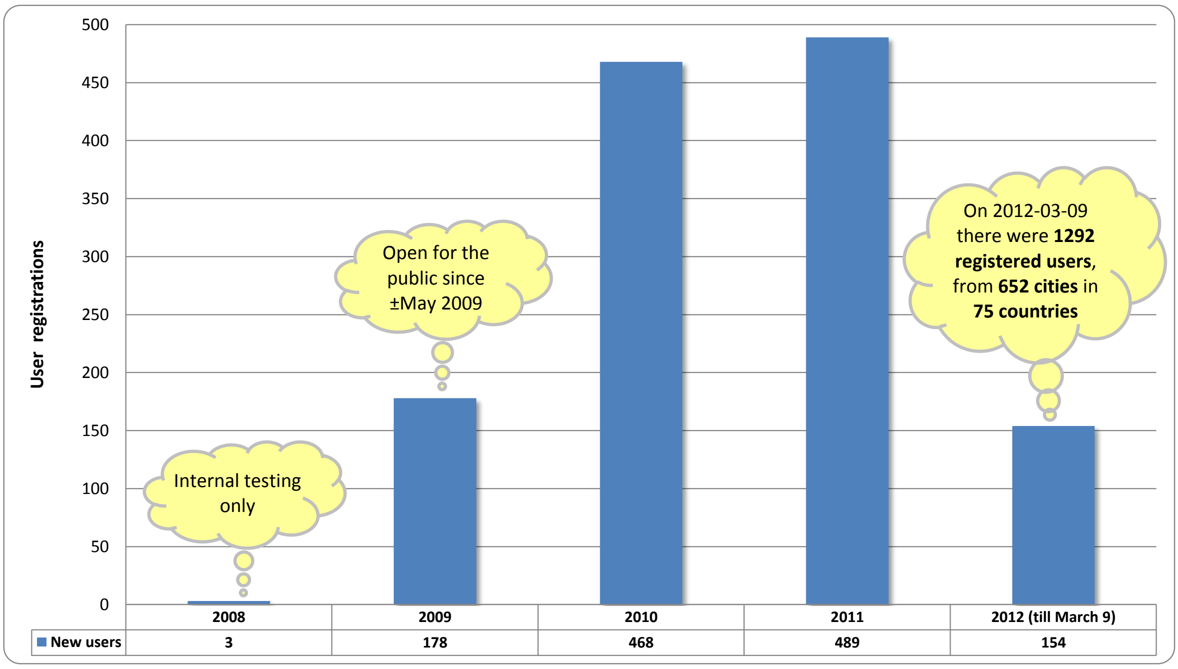


Chart 5.1: Newly registered NoiseTube users per year (data up to 2012-03-09)

[32] Most automatic taggings remain the same for long sequences of measurements. For instance, without tagged intervals all measurements of a track made between 7 and 8 AM would be individually associated with "morning", requiring as many tagging records as there are measurements in the track; whereas now this can be done with a single tagged interval and a single tagging record. However, we should note that we have not yet changed all NoiseTube CM code to work with tagged intervals.

Up to 2012-03-09, our users had submitted a total of 4243 tracks, consisting of 340 sound level measurements on average (stand. dev. 1482), made in 423 different cities. The cities are spread across 59 countries on all continents except Antarctica, as shown in figure 5.11, but most are concentrated in Europe, of which figure 5.12 shows a more detailed map.

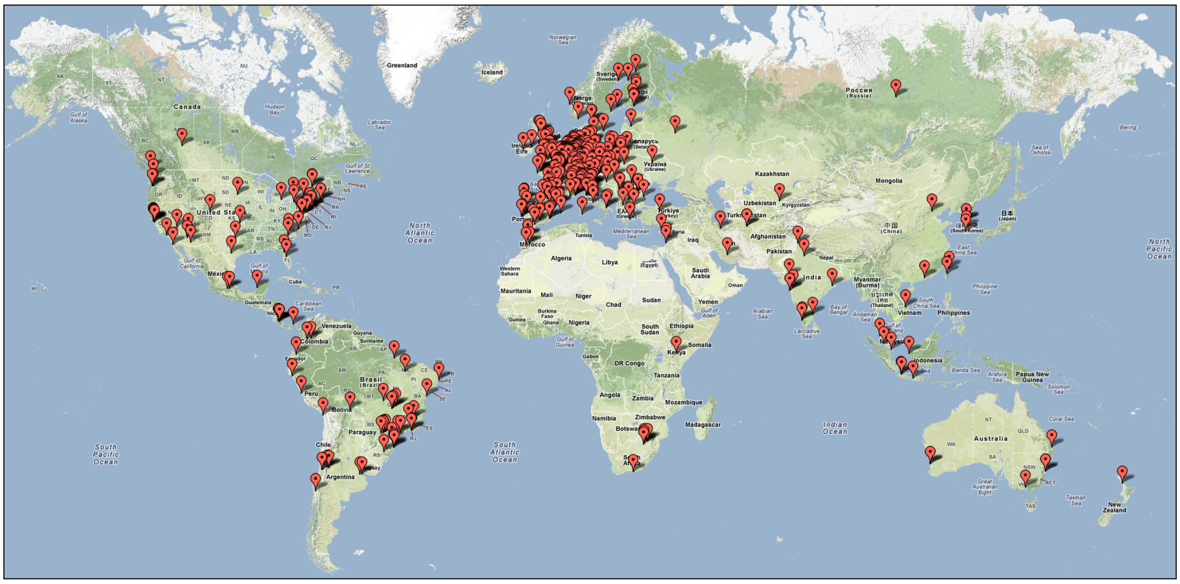


Figure 5.11: Map of cities with at least one track (data up to 2012-02-08)

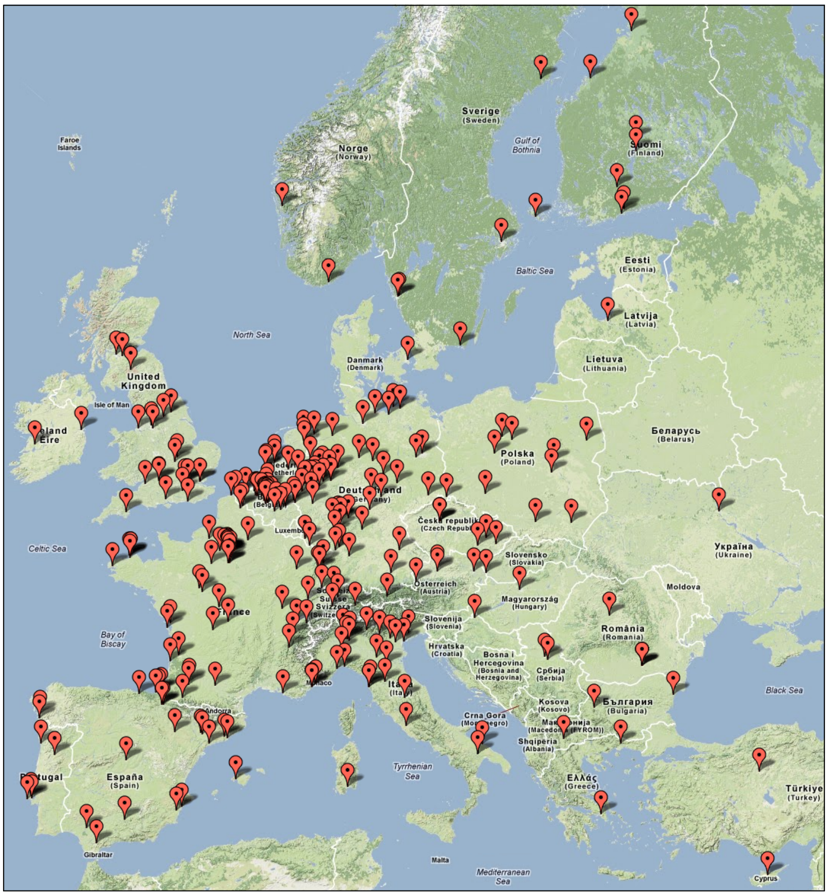


Figure 5.12: Map of European cities with at least one track (data up to 2012-02-08)

As illustrated by chart 5.2 tracks are not evenly distributed over cities: 122 cities only have 1 track, while the 6 most "popular" cities together have 37 % of the tracks. It is no surprise that Paris and Brussels, where NoiseTube has been extensively tested, demonstrated and used – by us, our colleagues and students, and a lot of volunteers – are together responsible for about 27 % of all tracks, made by 93 users. As illustrated by chart 5.3 the vast majority (81 %) of cities in fact only have one contributing user.

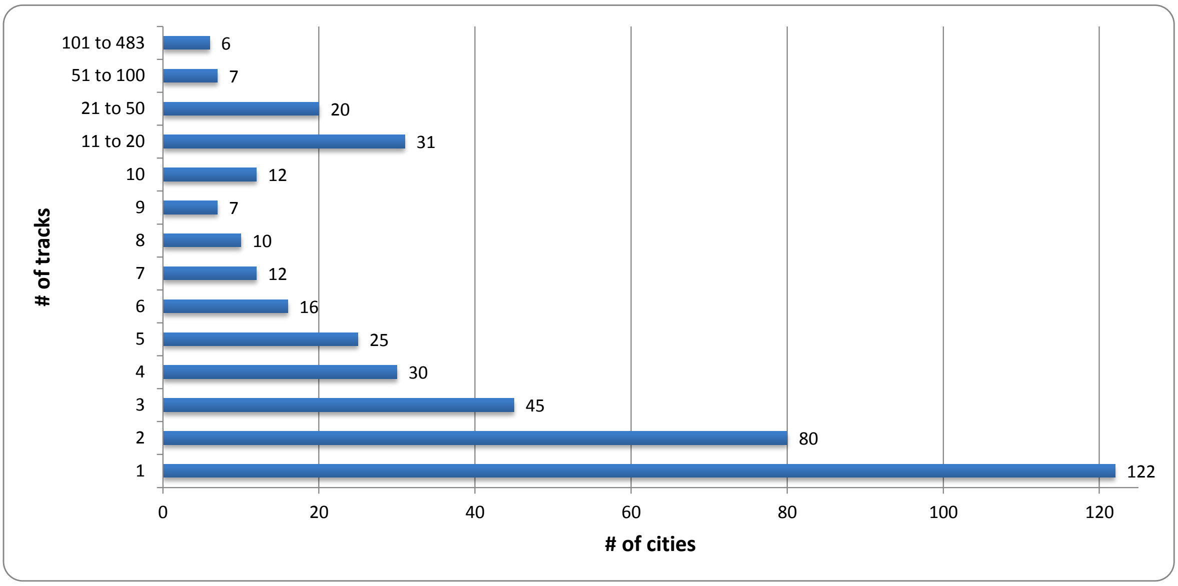


Chart 5.2: Number of cities with a certain number of tracks (data up to 2012-03-09)

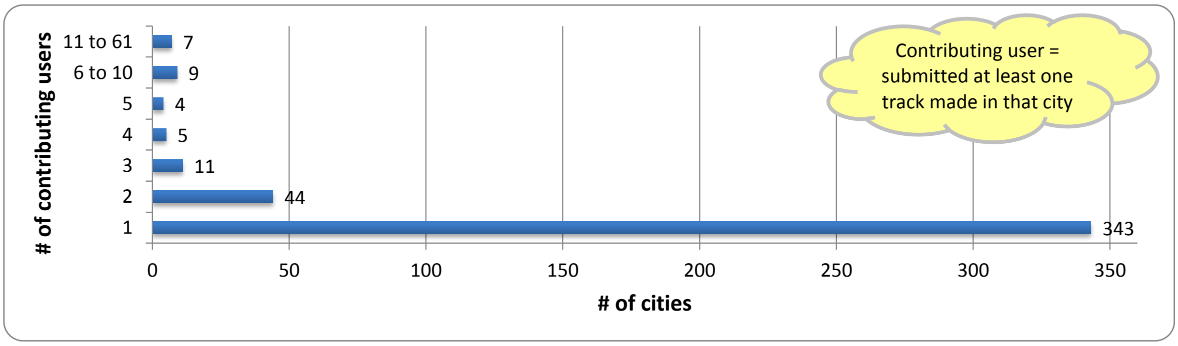


Chart 5.3: Number of cities with a certain number of users (data up to 2012-03-09)

As illustrated by chart 5.4 there are large differences in the degree of activity of our users. The 4243 tracks we have received up to 2012-03-09 were submitted by 642 users, or about 50 % of the total number of registered users. Consequently none of the other 50 % of our users has ever contributed a track, which is remarkable because contributing data is about the only functionality of the NoiseTube CM which really requires an account. So we could ask ourselves why those people bothered to register at all. One explanation could be that the website is not clear enough about which additional functionalities are available to registered users. Another interesting fact is that the 1 % most active users are responsible for 31 % of the tracks. As discussed in section 2.4.3.1, the 90-9-1 principle, a

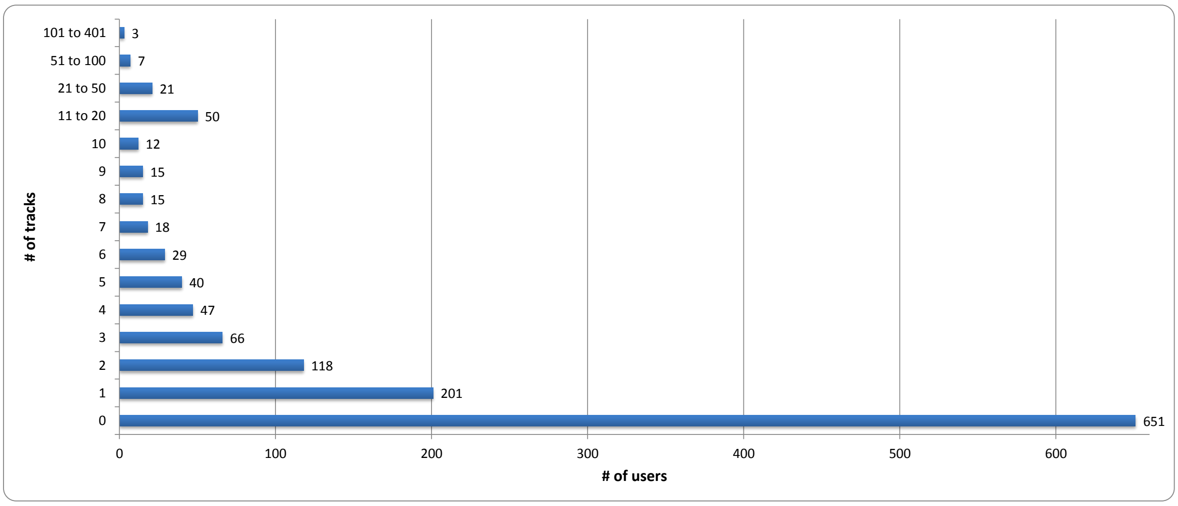


Chart 5.4: Number of users with a certain number of tracks (data up to 2012-03-09)

popular theory about participation in online communities, states that in a typical community 90 % of users are "lurkers" (who never contribute), 9 % contribute occasionally, and 1 % account for most contributions [370]. In our case this would be 50-49-1. However, note that we are only counting registered users, not all visitors of the website[33] – most parts of which can be consulted without an account – nor all NoiseTube Mobile users[34]. If we were to include all of those we would probably be closer to a 90-9-1 split. It is important to note that ±19 % of the tracks were submitted by NoiseTube/BrusSense team members – in other words more than 80 % of the data was *not* collected by us.

In the data received up to 2012-03-09, 137 users (21.3 % of contributors) have tagged 2866 measurements or intervals from 439 tracks (10.3 % of all tracks), for which they have used 338 distinct tags. To illustrate the relative popularity of different tags we generated the tag cloud shown in figure 5.13, based on data received up to 2012-02-05. To generate this tag cloud we have ignored tags that were entered for testing purposes. Since NoiseTube is used by people from all over the world it is no surprise that tags are entered in various languages. Besides English, which accounts for the biggest portion, other languages include Dutch, French, German, Italian, Portuguese and Spanish. While NoiseTube Mobile does not require or ask that tags are written in a particular language, the dominance of English was to be expected as its use is implicitly suggested due to the fact that the app, as well as the NoiseTube CM/website, is currently only available with an English UI, and the Java ME variant of the app has a list of example tags in English. In order to shed light on the types of tags and people's motivations to tag, we made a categorisation based on data received up to 2012-02-05. Chart 5.5 shows the result. For this chart and the percentages mentioned below we have ignored tags that were entered for testing purposes as well as a few that we were unable to make sense of.

33 About which we sadly do not have complete statistics.

34 Downloading and using the NoiseTube Mobile app does not require an account, only submitting data to the NoiseTube CM does. See chapter 6 for statistics on the number of app downloads.

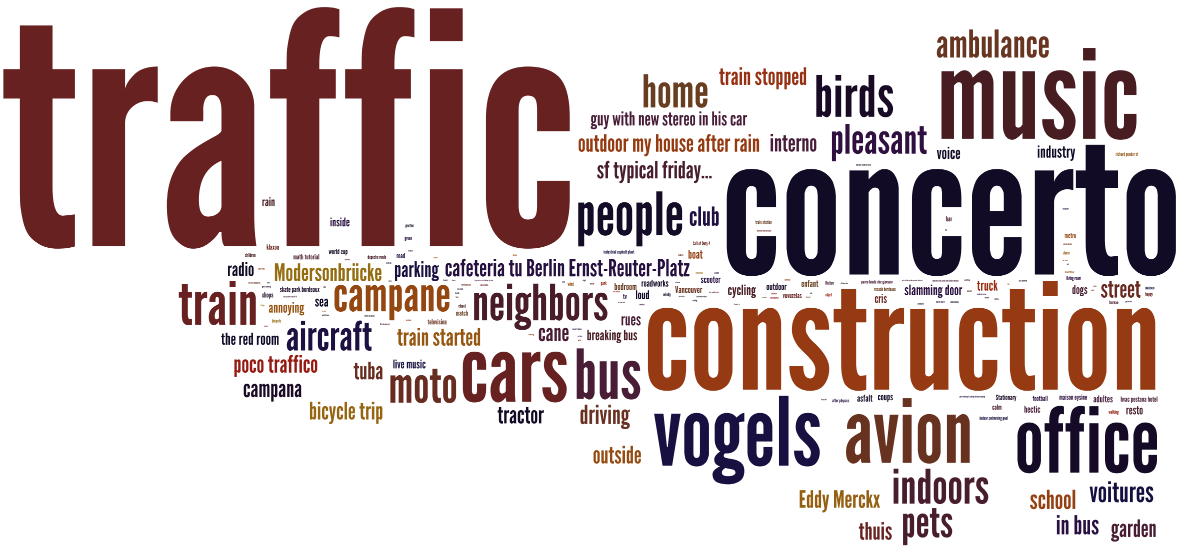


Figure 5.13: Tag cloud of social tags (data up to 2012-02-05)

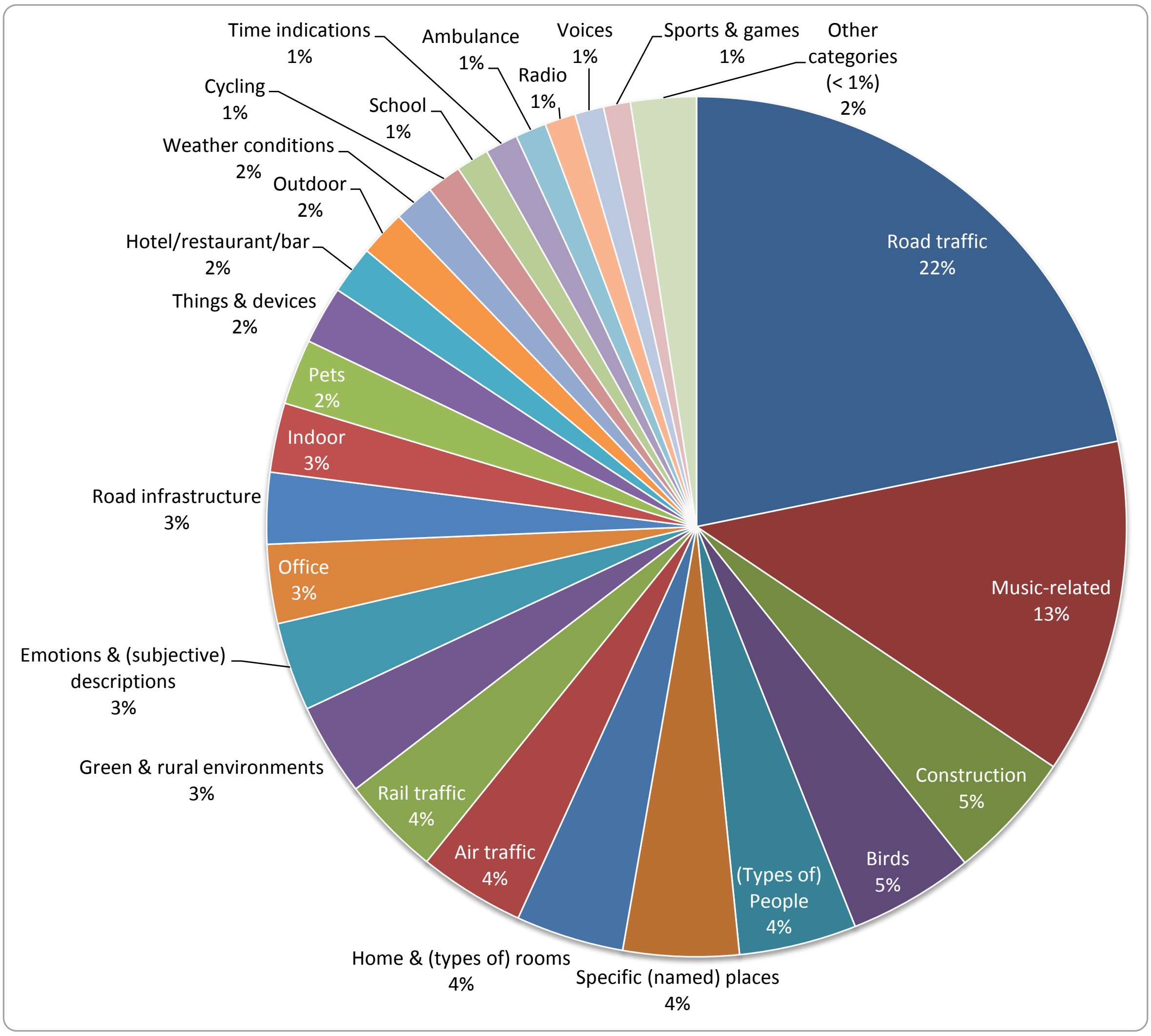


Chart 5.5: Social tag categories and relative frequency [%] (data up to 2012-02-05)

There are a few interesting observations to be made here – some expected, others less so:

- The main purpose we hoped (and suggested) tags would be used for is the identification of noise or sound sources. The data reflects this, as about 68 % of tag associations identifies a sound source. Looking at the types of sources we notice three of the typical sources of environmental noise, namely road, air and rail traffic, which together account for almost 30 % of all tag associations. What is more surprising is that very few people have mentioned industry as a source of noise (0.9 %), while many have reported on noise from roadworks or other construction activities (4.9 %). This is important because it indicates that, even though construction noise is ignored in the END legislation and hence in the resulting official noise maps (see section 4.3.2.7), our users do seem to be bothered by it, or at least feel the need to report it. This underscores the complementarity of our approach with respect to conventional methods for the assessment of environmental noise.

- Another expected and suggested reason to tag is the description of the context or circumstances in which measurements are made. The data shows that this has indeed been a common motivation to tag, accounting for ±30 % of associations[35]. This includes indications of place – both specific (e.g. "Modersonbrücke") and generic ("home", "office", "indoor", "outdoor"), time, activity ("cycling", "walking") and weather conditions. Note that these categories are similar to those produced by the automatic contextual tagging feature.

- Other expected/suggested reasons to tag are the description of the sounds themselves, the way they are perceived and the emotions they evoke. However somewhat disappointingly only 3.3 % of tag associations fall in this category. Examples of such tags are "annoying", "pleasant", "loud", "calm", "hectic", "still" and "happy".

- What we also hoped, but were less confident about, was that social tagging would not only be used as a channel for complaints and identifications of *noise* sources, but also as a means to express positive feelings and to identify any *sound* sources. The data shows that this is the case. Many users have reported on (presumably) pleasant experiences, such as going to concerts, hearing birds or watching sports. Among the few tag associations related to emotions and perception (see above) more than half are of a positive nature. This indicates that, rather than to focus solely on noise and its negative connotations, our users indeed consider it worthwhile to paint a more complete picture of the soundscapes they experience.

In general social tagging has not been used as much as we had hoped. One likely explanation is that the tagging functionality in NoiseTube Mobile, especially the Java ME variant, is not intuitive enough and hence takes too much time. In chapter 6 we discuss some tagging-related innovations that were recently added to the Android variant of the app and which will hopefully convince more people to start tagging or do it more often.

[35] Note that there is some overlap between the identification of sources and the description of context. For instance, weather related tags like "rain" and "wind" identify a sound source but also describe context.

5.7 Discussion

Here we discuss our experience with the deployment of the NoiseTube system and how that has influenced the system's design and the timing and choice of subsequent research steps. First, it is worthwhile to look back at the vision originally put forth by Steels [493–495] and extended in section 2.5. Therefore box 5.1 recapitulates three characteristics of the communities for which community memories are intended.

1. Community memories are intended for real communities of real individuals, not diffuse groups that flock anonymously through the Internet and have no real state in the management of a (shared) commons. Such a community is usually formed by fellow-citizens of a neighbourhood, village, town or city. These people are assumed to act in concert because they share a common goal.
2. In order to function as a community it is necessary that members recognise each other as individuals and that they meet face-to-face. Such meetings help to build trust and coordinate activities (e.g. data collection campaigns).
3. It is important that identity cannot be hidden and all actions can be traced back to the individuals who carried them out.

Box 5.1: Characteristics of communities as users of community memories

When the NoiseTube project got started we did not have an specific, existing group of volunteering citizens to work with. However from the start it was our intention to open up the system to a wide audience as soon as possible. While this is in line with our ambition to conduct applied research aimed at real-world deployment (see section 3.5), we also had three more specific reasons for it. First, we wanted our system to be tested by real citizens in their daily environments[36] in order to receive feedback (and bug reports) early on. Second, we wanted to show that the core technology (i.e. measuring sound level with mobile phones) works and spread word via our users and the media. This way we hope(d) to convince local grassroots organisations or authorities to try it out and eventually set up coordinated noise mapping campaigns – with or without our direct involvement – with groups of fellow citizens, much like described in box 5.1. Third, we were interested to find out whether it would be possible to attract a critical mass of (individual) users in certain places such that their efforts would cumulate into neighbourhood- or city-wide noise maps, possibly even without coordination. In posing this question we were strongly inspired by Web 2.0 success stories such as Wikipedia and the "blogosphere", in which people start contributing out of mainly individualistic motivations, but over time see the emergence of collaboration in support of the collective goals of the/a community[37].

[36] Rather than only our colleagues or students.

[37] See section 2.4.3.1 as well as our 2010 journal paper [334] (written in 2009).

This deployment strategy provides another argument for the design of NoiseTube as a system that supports participation at different, nested scales (see sections 3.5.1 and 5.3). Rather than developing a system solely aimed at groups as described in box 5.1, we made sure that individual users can try out the technology on their own. The data these individuals collect may or may not be submitted to the NoiseTube CM and there it may or may not be shared with others. While existing or newly formed groups can adopt the system for locally coordinated noise mapping campaigns, no such group was involved in the initial design and testing during 2008-2009. In terms of the sensing scales discussed in chapter 3, this can be seen as an effort to evolve from personal sensing to mass sensing, with or without instances of group sensing initiatives being active within the greater whole.

The mass sensing level applies to the system as a whole, with users distributed across cities all over the world. However, in those places where a critical mass of users would form – e.g. 100 individuals active in the same city – collective efforts, whether coordinated or not, can be seen as instances of mass sensing as well. As explained in section 3.3.3, at this level social ties and trust relationships among contributors tend to be weak and privacy concerns are likely to become an issue. This explains why, contrary to characteristic 3 in box 5.1, the NoiseTube system does allow users to hide their true identity[38], although all contributions remain associated with the submitting user.

To attract (potential) users we have actively engaged in a number of publicity efforts. We presented the project at various scientific and popular events and posted about it on Internet forums and blogs frequented by technology savvy "early adopters". Over time the project became better known and we also started to be contacted by popular online, print and audio-visual media[39]. As illustrated by the usage statistics discussed in section 5.6, this has enabled us to attract a fairly large community of users.

In its entirety – $\approx$1300 registered users, from 652 cities in 75 countries – this is clearly not a community as described in box 5.1. Indeed, in a sense these people *do* flock anonymously through the Internet, and while many probably share a concern about noise in their respective area, they are too spatially distributed to be considered stakeholders of a shared commons. Looking at the "city-level communities" (see chart 5.3) we found that there are currently only 7 places with more than 10 contributing users. Even within those cities there are relatively few areas where sufficient data has been collected to allow truly informative noise maps to be produced. This underscores the difficulty of recruitment and the importance of local, "offline" coordination (see challenges 2 and 3 in section 3.4).

In retrospect it might have been better to include communication features that enable NoiseTube users to contact one another through the CM website or the mobile app. This would likely facilitate coordination and possibly even the emergence of spontaneous

[38] Upon registration the CM only asks people for a username (which can be a nickname), an e-mail address, their hometown and their country.

[39] Refer to appendix G for an overview of all dissemination efforts and media mentions.

collaboration between strangers. Moreover, currently the NoiseTube CM does not allow users to create groups for local campaigns, which others could then explicitly join and which would make it possible to share data with group members alone rather than with everyone. Such a feature may facilitate coordination as well as reduce privacy concerns.

While such features will likely be added in the near future (see chapter 8), we have not done so earlier due to a change in focus. In 2010 we decided to first conduct a thorough evaluation of the existing system and the participatory noise mapping approach in general, rather than to continue adding more features which may or may not be used by (until then) mostly anonymous users. This work, which we discuss in detail in chapter 7, involved both laboratory and real-world experiments. In the former we collaborated with professional acousticians to evaluate and improve the accuracy of the sound level measurements made by the NoiseTube Mobile app. In the latter we worked with an existing group of volunteering citizens to set up noise mapping campaigns coordinated by us. In this manner we wanted to evaluate how the system performs in practice (in terms of usability and data quality) as well as to gain insight in the organisational aspects of such campaigns and how they can be supported through technology and best practices.

There are three main reasons for this shift in focus. First, it allows us to study a situation that is much closer to the initial community memory vision and the group sensing scale[40]. Second, coordinating campaigns ourselves and taking control of more parameters[41] is a more direct route to reproducible, comparable and (if they meet certain standards) credible results – providing us with crucial arguments to convince citizens' organisations and authorities of the potential of NoiseTube. Third, the hands-on experience gained in real-world experiments enables us to make informed choices about improvements, extensions or adaptations of the system needed for specific types of users and/or contexts.

In the real-world experiments discussed in chapter 7 we have taken the perspective of bottom-up, citizen-led initiatives (assisted by professional scientists). As explained in section 5.3 the initiative for locally coordinated noise mapping campaigns can also come from authorities (i.e. top-down). In chapter 8 we discuss concrete plans to study participatory noise mapping from the perspective of authorities in the near future. Taking both perspectives is important because system requirements are likely to be different. For instance, authorities will likely require us to produce data in specific formats. The direction of initiative may also affect the concerns of campaign participants. For example, citizens may be less inclined to share potentially sensitive data (i.e. about personal whereabouts) with a faceless authority and fellow participants they do not know, than with members of the same citizens' organisation who they presumably know and trust[42].

[40] See section 2.5.1 (partially summarised in box 5.1), section 3.3.3 and section 3.5.1.

[41] E.g. the choice and calibration of devices, and the measuring protocol (see chapters 6 and 7).

[42] Although the opposite is also conceivable.

Although there is currently only one instance of the NoiseTube CM, since the release of the source code in June 2011 other parties are free to set up a separate and possibly adapted system. As far as we know this has not happened yet, but in the future we may help interested organisations to do so. Giving specific organisations "their own" NoiseTube system would be a more scalable alternative to the abovementioned group feature. Scalability is an important concern because, as a small research team, our resources – in terms of manpower, funds and (server) infrastructure – are limited. It is noteworthy that as the number of CM users increased we have already experienced performance issues. We have partially remedied these growing pains by means of data model tweaks (see section 5.5.2 for an example), caching, nightly and hourly batch processing[43] and eventually migration to a faster machine. However, if the number of users were to rise from hundreds to (tens of) thousands it may still become infeasible to manage the NoiseTube service without additional manpower (apart from technical improvements)[44].

	Advantages	Disadvantages
Centralised **Single NoiseTube CM instance**	Opportunity for (implicit) collaboration between strangers (so far largely unrealised) Central access to all collected data (useful for research purposes) Low barrier to try out the system or organise campaigns (organisations do not need to set up their own server)	Limited scalability Too generic ("one size fits all") Too open (privacy concern)
Decentralised **Multiple NoiseTube CM instances**	More scalable Opportunity to adapt the system for specific contexts (including the needs and abilities of specific users) Reduced privacy concerns (may be closed to outsiders)	Fragmented datasets Risk of fragmentation or "forking" of the codebase Citizens' organisations may lack the technical knowhow and funds required to set up and manage a server

Table 5.2: Advantages and disadvantages of one vs. many NoiseTube CM instances

In table 5.2 we summarise the pros and cons of the centralised deployment strategy we have initially followed (i.e. a single NoiseTube CM instance for everyone) versus a decentralised one in which organisations would set up their own separate instances. We should note that these strategies are not mutually exclusive. We should also clarify that, while the LGPL license [195] obliges others to release any changes under the same terms, the integration of valuable improvements in the main codebase could require a lot of work (and perhaps may never happen at all), hence the risk for fragmentation or "forking". Clearly, both strategies have advantages. However, in order to move forward in our quest to put mobile sensing and community memories in everyday practice (see section 3.5) we are inclined to think that shifting to a decentralised model is more likely to be successful, as well as closer to the vision put forth in section 2.5. Hence in the future we may work with communities and/or authorities to set up separate, customised CM systems. At first these would probably be hosted on our server(s), alongside the central NoiseTube service, but over time we could simplify and document the deployment (i.e. server set up) procedures such that the barrier for fully independent initiatives is lowered.

[43] For instance to generate KML maps ahead of being requested.

[44] Also see section 3.3.1.2.

5.8 Conclusion

In this chapter we first situated the NoiseTube research project, which started in 2008 at Sony CSL Paris and continues at VUB since 2010. Then we introduced the NoiseTube approach, our innovative solution to the assessment for the environmental noise by both citizen- and authority-led initiatives. This approach, which is enabled by the NoiseTube system, constitutes a cheap, participatory and people-centric alternative or complement to conventional noise assessment methods. Next we presented the architecture of the NoiseTube system itself and the functionality and implementation of the NoiseTube Community Memory, the server-side component.

In our discussion of usage statistics we showed that our system is used by people from around the world, although these are, for the most part, lone individuals rather than tightly-knit communities. Moreover, there is a similar degree of participation inequality as in typical online communities. We also showed that the social tagging feature is being used for the purposes we had in mind, yet not as frequent as we hoped. Next we reflected on our initial deployment and recruitment strategy and contrasted it with our vision regarding community memory systems and the communities that use them. Here, the main thing to remember is that, in absence of an existing group of volunteering citizens, we first followed a Web 2.0-type strategy, resulting in a global, online community, rather than a local, "real" one. However, in doing so we have been able to test the software on a large scale and attract (media) attention, which in turn eventually enabled us to convince a group of grassroots activists to help us put participatory noise mapping to the test in a coordinated campaign. The result of that collaboration is discussed in chapter 7.

Regarding the NoiseTube CM software the main message is that it is neither a prototype, nor a final "product". Rather its feature set is the result of a balancing act between the need to provide a usable, operational system for users who were largely unknown to us, and the need to have a platform[45] for experimentation and demonstration purposes. In the future we will likely make a number of general improvements and extensions to the software (e.g. communication and group features) and/or work with others to adapt it to suit the needs of specific communities and campaign initiative takers.

The NoiseTube system and the participatory noise assessment method it enables form the most important contributions put forth in this dissertation. This discussion is continued in the next two chapters. First, in chapter 6 we take a detailed look at the functionality, design and implementation of the NoiseTube Mobile application, the client-side component of the system. Moreover, chapter 6 treats the main examples of related work. Then, in chapter 7 we discuss the validation of the NoiseTube approach and system by means of experiments in the lab and coordinated participatory noise mapping campaigns in the field.

[45] Consisting of both a running system and an incoming stream of "real" citizen-collected data.

Chapter 6

The NoiseTube Mobile application

6.1 Introduction

In this chapter we take a detailed look at the functionality, design and implementation of *NoiseTube Mobile*, the client application of the NoiseTube mobile sensing system. This freely downloadable participatory sensing app turns a mobile phone into a personal, portable, low-cost sound level meter. In terms of the sensing scales discussed in chapter 3, the app is designed to be used as a client for personal, group or mass sensing. This means users are free to share their data – series of (geo-)tagged, time-stamped sound level measurements – through the/a NoiseTube Community Memory, or to keep it to themselves (or share it by some other means). Additionally this chapter discusses the main examples of related work – with respect to NoiseTube Mobile or the NoiseTube system as a whole.

In section 6.2 we first introduce three variants of NoiseTube Mobile: two of which are publicly available, one for the Java ME platform and another for the Android platform, and a third one, for the iPhone, which will be released very shortly. In the next sections we focus solely on the first two variants. In section 6.3 we take the perspective of the user and describe the different functionalities, including some experimental ones, and discuss some important usability concerns. Next, section 6.4 explains why Java ME and Android were chosen as the initial platforms and documents the dependencies the apps for both platforms rely on. Then, section 6.5 thoroughly motivates and documents the design and implementation of both apps, which share a significant part of their source code. Related work from within and outside academia is discussed in section 6.6. Finally, section 6.7 wraps up the chapter.

6.2 Variants

There are currently two fully fledged variants of the NoiseTube Mobile application and a third one is about to be released. Each of these variants targets a specific software platform found on commercially available smartphones.

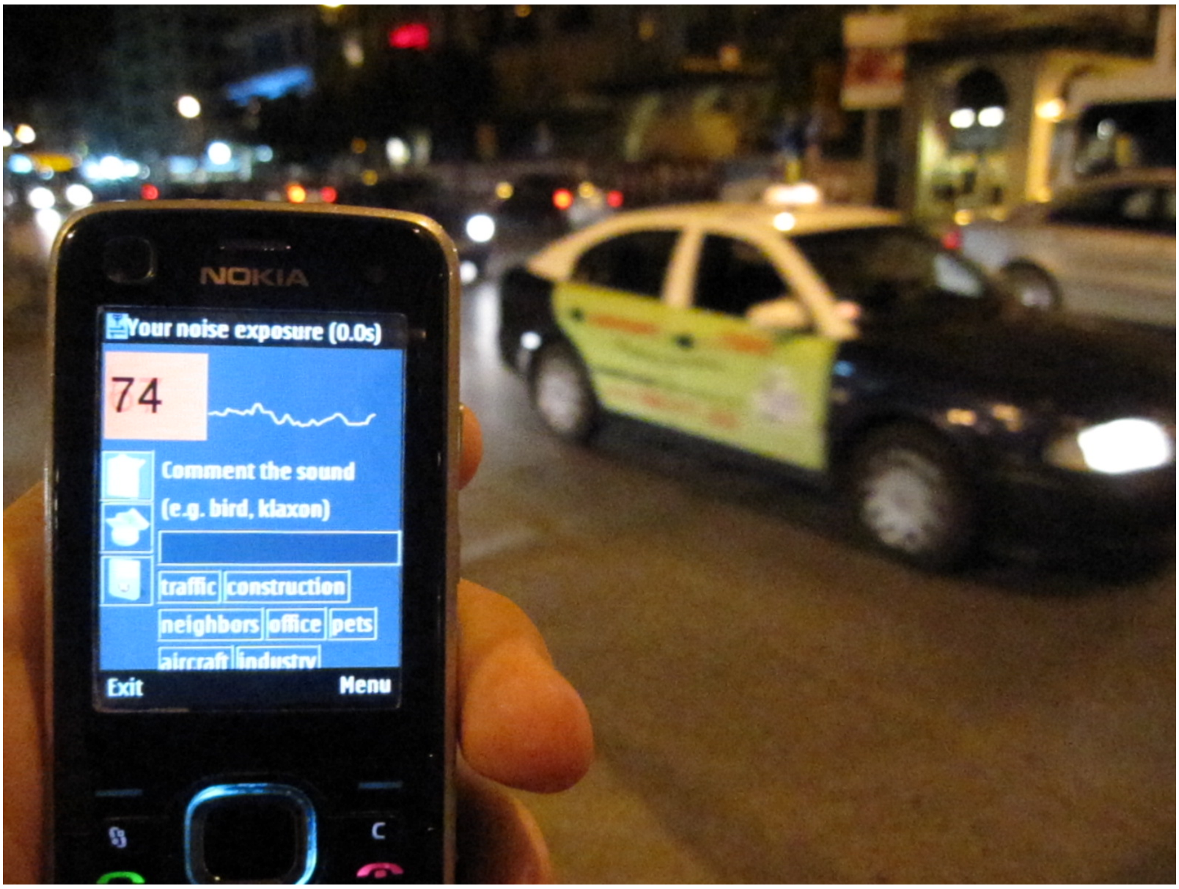


Figure 6.1: An early iteration of *NoiseTube Mobile for Java ME* being used in a busy street in Thessaloniki, Greece (May 2009)

The initial variant of NoiseTube Mobile was developed for the *Java ME CLDC/MIDP platform* (*Java ME* for short). Work on this application began in September 2008[1] at Sony CSL Paris and it was first released to the public in May 2009. Since then, updates have and continue to be released at a varying rate. Figure 6.1 shows this application in action. Also in spring 2009, the app, as well as the NoiseTube system as a whole, was first presented to academic audiences at a workshop [335] and two conferences [331, 332]. In 2010 it was covered in our first journal paper about NoiseTube [334].

In October 2010, the development of a variant for the *Android* platform began at VUB. The first publicly available version was released in June 2011 and updates continue to be

[1] After work on the NoiseTube Prototype had been completed (see appendix C).

released regularly. The Android application contains almost all functionality found in the original variant as well as some new features, as we will discuss in section 6.3.

NoiseTube Mobile variant	From	To	Number of downloads by unique IP addresses	Downloads/day
Java ME	2010-02-28	2012-05-29	8376	10.20
Android	2011-06-01	2012-05-29	3876	10.68
Total	2010-02-28	2012-05-29	12252	14.92

Table 6.1: NoiseTube Mobile download statistics[2]

Both apps can be downloaded for free by anyone, respectively from the NoiseTube website [375] and the Google Play app store [377]. People do not need to register on the NoiseTube website in order to download or use the apps. This is only required for uploading data to the website (i.e. the community memory). Table 6.1 shows statistics about the number of times the apps have been downloaded so far. In June 2011, upon receiving formal approval of Sony France, we made NoiseTube an open source project. The source code of both NoiseTube Mobile apps – and that of the NoiseTube Community Memory – was published on Google Code [378] under the terms of the LGPL v2.1 license [195].

Apart from the two NoiseTube Mobile variants released so far, a third one, targeting the iPhone – or rather its *iOS* platform – was developed at VUB in 2011 and is currently under review by Apple for release on the *iTunes* app store [31][3]. Furthermore, together with students working under our supervision, experimental derivatives of NoiseTube Mobile have been developed, which we will discuss in chapter 8.

In the next sections, we will focus on the first two variants, which share a significant portion of their code. The full name of the first is *NoiseTube Mobile for Java ME* and that of the second is *NoiseTube Mobile for Android*. In what follows we will often simply refer to them as respectively "the Java ME application" or "the Java ME variant" and "the Android application" or "the Android variant". Our description of the functionality, design and implementation of this software is based on the current state of the code. Hence, we will highlight a few new features and refactorings which are still being tested or are just not released yet, but which will be part of the next update for either application[4].

[2] Sadly, we have lost the log files of the NoiseTube website from before 2010/02/28, therefore we do not know how many times the Java ME app was downloaded since its initial release in May 2009.

[3] In accordance with the LGPL v2.1 terms, the source code of this iOS port will be released under the same license soon after the app is made publicly available.

[4] Concretely, the discussed versions of the Java ME and Android application are respectively v2.2.0 and v1.3.0, both of which will probably be released in the summer of 2012.

6.3 Functionality

Before we dive into the details of the design and implementation of NoiseTube Mobile, we look at it from the perspective of the user. This section describes and illustrates the functionalities offered by both, or either one of, the Java ME[5] and Android variants.

In line with the approach motivated in section 3.5, we designed NoiseTube Mobile as a participatory [71], rather than an opportunistic [74], sensing application, and we aimed to support scenarios at personal, group *and* mass sensing scales. As a consequence of the choice for the participatory sensing paradigm, we expect and support the active involvement of the user in the sensing process. Concretely that means users of NoiseTube Mobile are in full control of when, where and for how long they make measurements and whether or not they comment on the collected data by means of social tagging. By opting for a multi-scale approach, we aim to make the application useful and interesting at a personal level (i.e. for individual citizens) and for groups (i.e. communities collaborating to measure noise in their local area), while keeping in mind that individuals and groups may want to contribute to a larger mass sensing project. The fact that the number of app downloads (see table 6.1) is much higher than the number of registered users on the NoiseTube CM/website (see section 5.6) proves that many people have indeed chosen to use or try out NoiseTube Mobile on their own as a personal sensing application.

6.3.1 Sound level measuring

The primary function of NoiseTube Mobile is of course to measure the sound pressure level (SPL, see section A.2.3) in the immediate surroundings of the device and the user. Basically our app turns mobile phones into personal, highly-portable sound level meters.

As we explain in section A.5.1, there are two main kinds of sound level meters (SLMs): conventional and integrating-averaging ones. The principal difference between both is the way they average sound pressure over time (see section A.5.1.3), resulting in two ways to measure SPL: *time-weighted sound level* – measured with conventional SLMs – and *time-average sound level*, better known as *equivalent continuous sound level* or simply L_{eq} – measured using integrating-averaging SLMs. Today, acousticians generally prefer the latter type of device when they need to assess noise exposure over certain periods of time. The EU's Environmental Noise Directive (END) [173] (see section 4.3.2.7) also specifies that exposure should be expressed as L_{eq} (and derived) values. Although integrating-averaging SLMs are much more expensive than the conventional kind (see table A.5 on page 293) the former's functionality is not really harder to implement than the latter's, at

[5] A more user-oriented description of the Java ME app (including installation instructions) can be found in the *NoiseTube User Guide* [496], although it is no longer up-to-date with the current version.

least on a device that can record and process digital audio signals. For these reasons we have chosen to make NoiseTube Mobile act like an integrating-averaging SLM[6], producing series of "short L_{eq}" values[7], each taken over a 1 second interval. The application does this by repeatedly recording 1 second-long pieces of digital audio through the device's built-in microphone. Each piece of audio is analysed on the phone by means of a real-time digital signal processing algorithm, resulting in a single $L_{eq,1s}$ value.

Like actual SLMs, NoiseTube Mobile also applies frequency weighting. As explained in section A.5.1.2, frequency weighting serves to adjust the measured SPL based on the frequencies present in the sound, in order to account for the frequency response of human hearing (see section A.3.4.1). The de facto weighting for general purpose use, supported by all commercial SLMs and mandated by most noise assessment regulations (including the END [173]) is *A-weighting*. Therefore, we have implemented a digital A-weighting filter in NoiseTube Mobile. Hence, the apps actually measure A-weighted equivalent continuous sound level over 1 s intervals, denoted as $L_{Aeq,1s}$ and expressed in dB(A).

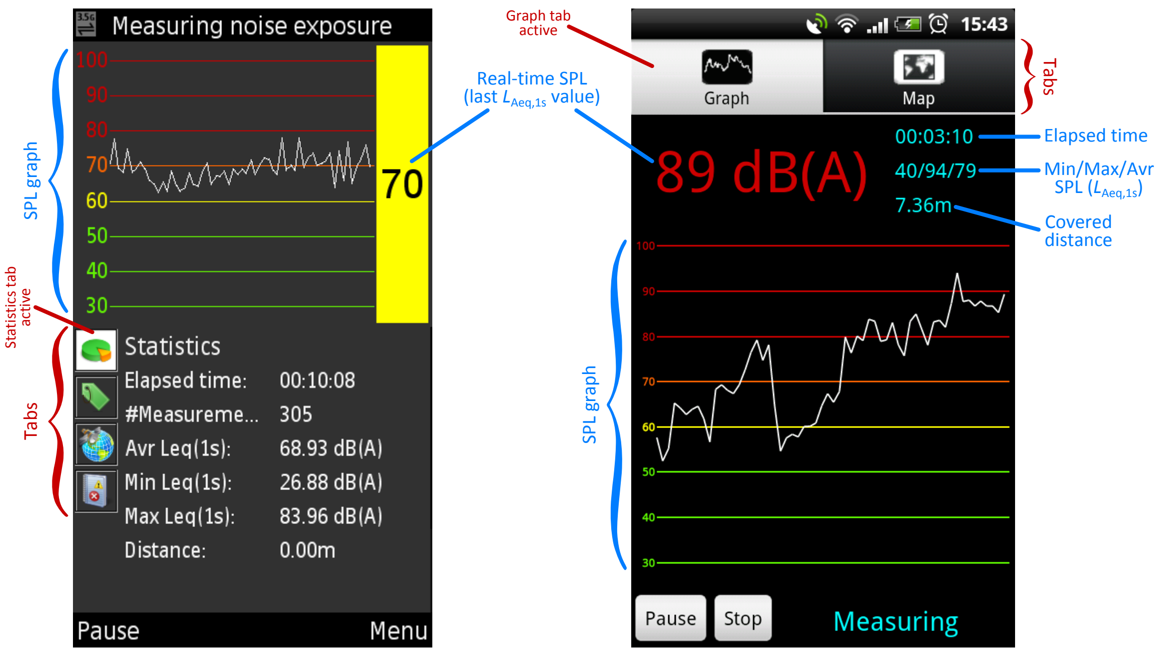


Figure 6.2: Measuring sound pressure level with NoiseTube Mobile (Java ME variant on the left, Android variant on the right)

As mentioned in section 5.4, we refer to a consecutive series of measurements made by a single user as a *track*. When the user starts the NoiseTube Mobile app it automatically starts measuring and adds the measurements to a newly created track. Each measurement is associated with a timestamp, representing the exact date and time it was made. While measuring, the application appears as shown in figure 6.2. The measured SPL is

[6] Recently, we also implemented an algorithm to measure time-weighted sound level, as conventional SLMs do. We may release this as part of a separate "multi-mode" SLM app for more experienced users.

[7] The advantage of short L_{eq} series is that, when the values are taken of contiguous intervals, they can be easily averaged to calculate the L_{eq} over longer periods of arbitrary length (see section A.5.1.3.2).

numerically displayed in real-time at the top of the screen, and the last 60 measurements are shown on a graph. To make the decibel values easier to interpret, both the real-time display and the graph use a colour scale ranging from green (for low sound levels) to red (for high levels). We should note that the "average" $L_{\mathrm{Aeq,1s}}$ value shown in the interface (see figures 6.2 and 6.3) is computed as a logarithmic composite (see equation A.27 on page 303) and thus represents the "overall $L_{\mathrm{Aeq},T}$", in which T is the duration of the track.

At any time the user can stop measuring by hitting "Stop", at which point the current track is ended. In the Android app the user is then presented with a summary of the track, as shown in figure 6.3. When hitting "Start" measuring begins again and measurements are added to a newly created track. In the interest of transparency and privacy (see sections 6.3.6.1 and 6.3.6.2), users can temporarily interrupt measuring by hitting "Pause". When they later resume measuring, new measurements are added to the same track.

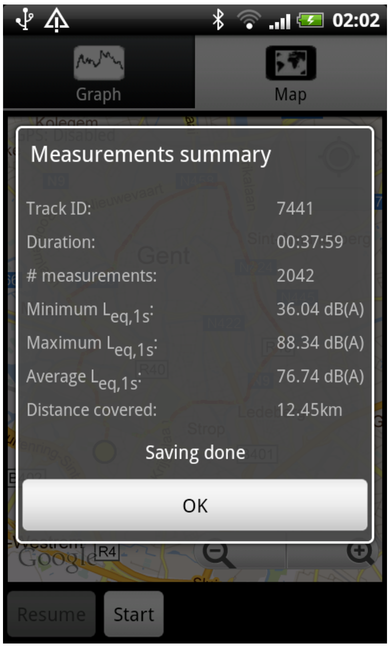


Figure 6.3: Track summary shown upon ending a track (Android variant only)

Like a real SLM, NoiseTube Mobile can and should be calibrated to achieve the best possible accuracy. Because different mobile phones have different microphones (along with other properties), the software must be calibrated for each particular model. The apps come with built-in settings for a number of models – currently 12, more will be added soon – and automatically download new ones from the NoiseTube website/CM [375] whenever we make them available. Every time they are started the apps compare the brand and model of the device with the list of available settings and select the most fitting one[8]. To interpret measurements it is important to know whether they were made on a calibrated device, therefore information about the used setting is stored/transmitted along with the data. Additionally the Android app also informs the user about whether or not a fitting calibration is used when a new track is started. In the Java ME app we used to offer a user-accessible calibration mode through which users could manually adjust the calibration setting[9]. Later, we developed separate *Phone Tester* apps (one for each platform) to test and calibrate new models. These are mainly intended for internal use, but upon request we make them available to other interested parties. Hence, the user-accessible calibration mode is no longer part of the current Java ME app, although in the future we might (re)introduce something similar in both NoiseTube Mobile variants.

[8] When there is no match for the exact model, a calibration for another model of the same brand is taken, and when there are no settings from that brand an overall default setting is used.

[9] Instructions to do so are described in [379].

6.3.2 Geo-tagging & map view

To make the collected data useful for others, and possibly also for personal usage, it is usually necessary to know exactly where measurements were made[10]. Therefore, NoiseTube Mobile can automatically geo-tag each measurement – i.e. associate it with geographical coordinates[11]. This information is stored/transmitted together with each measurement. To do this the apps rely on GPS – usually through a built-in receiver, although on some devices an external (Bluetooth-connected) receiver can be used. Even though today's smartphones usually include other positioning technologies[12] we do not use these because they are generally less accurate[13]. Figure 6.4 shows the geo-tagging related interface of both NoiseTube Mobile variants. In the interest of privacy, and possibly to lower power consumption, users can temporarily or permanently disable GPS at any time[14].

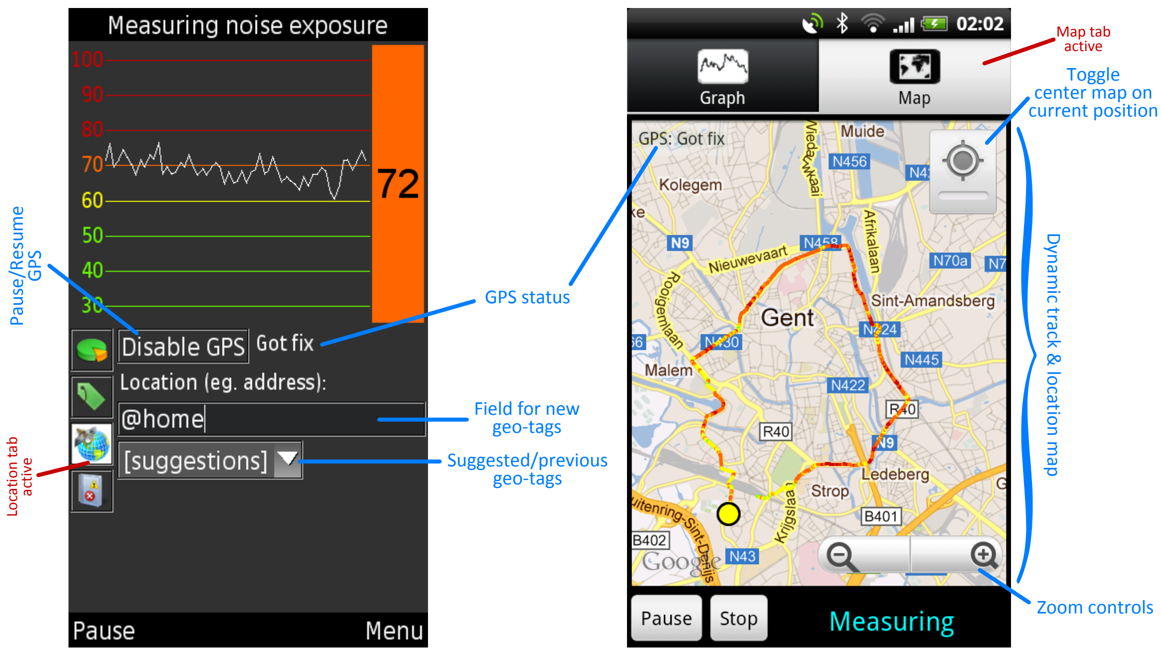


Figure 6.4: Geo-tagging in NoiseTube Mobile
(Java ME variant on the left, Android variant on the right)

As shown on figure 6.4, besides relying on GPS, the Java ME app also allows users to manually geo-tag measurements by typing a city or a street name, a full address, or a

[10] Especially when the purpose is to create accurate noise maps.

[11] With respect to the WGS84 datum [597].

[12] Based on triangulation w.r.t. cellular network antennae and/or Wi-Fi access points [53, 114, 252, 448].

[13] In fact, some smartphones combine multiple positioning technologies at the level of the operating system – to lower response times, reduce power consumption and increase accuracy. So in a sense we do rely on non-GPS positioning technologies, but never on their own.

[14] In the Java ME app there are two ways to disable GPS, a pause option (shown on figure 6.4), and a permanent one in the preferences screen of the app (see figure 6.7). The only difference is that pausing GPS is not remembered the next time the application is started. In the Android app GPS usage must be disabled in the preferences screen (see figure 6.7).

"semantic" descriptor (e.g. "home", "work", "school", etc.). We had three reasons for this. First, when we started with the NoiseTube project, built-in GPS receivers were still a fairly exotic feature, only found in the most expensive smartphones[15]. Second, we intended to provide an acceptable alternative to users who refuse to disclose their exact location due to privacy concerns[16]. Third, we recognised that, when assessing noise exposure in certain places or at certain times, knowing in what kind of place it is heard – e.g. at home vs. at work – may be just as (if not more) important than knowing the exact coordinates. The Android app does not include a similar feature. The reason is that users of the Java ME app rarely used it, and shared data without any geo-tags is often essentially useless to the community. Hence, we want to promote geo-tagging by GPS. Fortunately, all Android phones come with a built-in GPS receiver, and nowadays many people have grown more accustomed to disclosing their location in return for services. Of course, the Android app still allows users to disable GPS at any time[17]. Moreover, manual geo-tagging is in fact still possible via the general-purpose social tagging feature (see below) – e.g. in figure 6.6 we see measurements being tagged with "@home".

In the Android app, every geo-tagged measurement of the current track is projected as a coloured circle[18] on a map, as shown in figure 6.4. The most recent measurement is drawn as a bigger circle with a black outline. Users can zoom in and out and pan around the map using touchscreen gestures. Because the Java ME platform does not provide a practical way to include a map view in an application the Java ME app lacks this feature.

6.3.3 Social tagging

As discussed in sections 2.5 and 3.5 and first proposed in [494], social tagging is an essential aspect of our community memory vision and an integral part of the representation building process in which users of a community memory implicitly or explicitly take part. Hence, NoiseTube Mobile lets users tag noise measurements. Typical – and suggested – motivations to do so are the identification of sound sources (e.g. "car", "plane"), the expression of subjective perception or personal opinion (e.g. "loud", "quiet", "annoying", "pleasing"), or the description of the context, which may include places, times or activities (e.g. "@home", "lunchtime", "eating"). However, because we do not restrict the tagging vocabulary in any way, users are free to use tags for any other purpose. Generally speaking, tagging thus serves to augment the objective, numeric decibel measurements with a

[15] In fact, for a while we even offered a separate "GPS-less" derivative of the Java ME app, which only included manual geo-tagging. This version was discontinued in November 2010.

[16] In a sense this can be seen as a manual version of the automatic location obfuscation techniques discussed in mobile sensing challenge 6 on page 70.

[17] In fact, in the future we plan to integrate a much better and more secure solution for the protection of personal location traces (see section 6.3.6.2 and chapter 8).

[18] Using the same colour scheme as the graph and the real-time SPL display.

human dimension. As discussed in section 5.5.1, in the NoiseTube Community Memory tags facilitate the interpretation[19], exploration and structuring of the aggregated data.

It is interesting to note that, although algorithmic solutions exist [118, 327], humans still significantly outperform computers in identifying, separating or classifying sound sources. Hence, and because we chose to design NoiseTube Mobile as a participatory sensing app rather than an opportunistic one, we felt it was only natural to leave this task to the user by letting him/her act as a "human sensor" (see sections 3.3.1.1 and 3.3.4).

The tagging interface of the Java ME app, shown in figure 6.5, only supports tagging one measurement[20] at the time. Users do so by typing one or more tags, or by selecting one from a list of suggested and previously used tags. The tag(s) are then associated with the measurement that was made 4 seconds before the user started typing/selecting them. Sadly, practical evaluation has shown that this approach does not work very well. One reason is that users often want to use tags to signal short incidents or events (such as an ambulance passing by), yet by the time they have started to type/select tags the event is usually already over. While the 4 s delay compensates for this[21], often the wrong value is tagged. Another is that users often want to tag a series of measurements at once rather than just one, since even such short events usually last longer than 1 second. In fact, we had foreseen this from the beginning, which is why the *Mobile Noise Tagging* prototype app (see appendix C) allowed users to indicate exactly which moment or period in time, and thus which measurement(s), they intended to tag. However, because the tagging wizard interface[22] was deemed too complex and demanding in terms of effort/time, this idea was not carried over to NoiseTube Mobile.

Figure 6.5: Social tagging in NoiseTube Mobile for Java ME

On Android, we have made social tagging much more intuitive by using the touchscreen, as illustrated by figure 6.6. By making a dragging gesture across the screen the user selects part of the graph – i.e. a continuous interval of measurements – after which a tagging dialog pops up (step 1). This solves both problems: the user can now pinpoint

[19] For instance by adding them to generated noise maps.

[20] Afterwards drawn as a blue vertical line on the graph.

[21] This is why we added the delay in v2.1.0 of the Java ME app; the 4 s was chosen as a best guess.

[22] See figure C.2 on page 313.

exactly which measurement(s) he/she wants to tag[23]; and he/she can tag multiple measurements at once. Also, users can now use a combination of newly typed tags and one or more earlier tags (steps 2–4). The resulting *tagged interval*[24] is then marked in blue on the graph. This works on any Android device because they all have touchscreens. Nowadays, many new Java ME devices have a touchscreen as well, but because some, and most older ones, do not, and as the platform's popularity is declining (see section E.2.2), we will probably not take the effort to add this feature to the Java ME app.

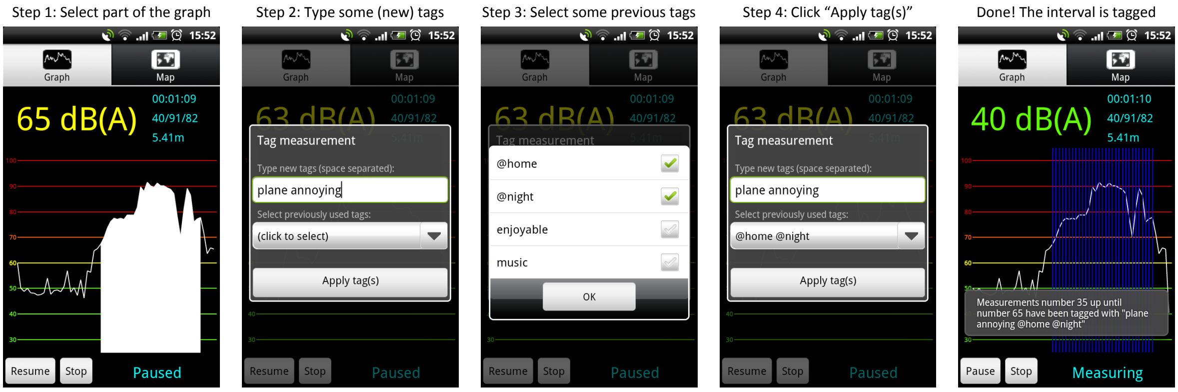


Figure 6.6: Social tagging in NoiseTube Mobile for Android

6.3.4 Automatic tagging

Besides social tagging, the NoiseTube system also supports *automatic tagging*, which is done by machines instead of people. Most of this happens in the community memory (i.e. on the server) as discussed in section 5.5.1.1, but NoiseTube Mobile also includes a mechanism to apply automatic tags[25] to measurements or intervals. For instance, the Java ME app can automatically detect periods of high exposure – 10 consecutive measurements above 80 dB(A) – or high variation – changes of more than 20 dB in less than 2 seconds – and tag such measurements with respectively "exposure:high" or "variation:high". These automatic tags are then displayed on the graph as red vertical lines.

In fact these two features were never more than a proof of concept and the detection algorithms are fairly primitive. Therefore they are not activated in the Android app (although their implementation is part of the shared source code), and may be removed in the next update of the Java ME app. However, in the future it would be possible to add more sophisticated automatic tagging features to NoiseTube Mobile without much work.

[23] As long as they are still shown on the graph, which gives the user about a minute to respond.

[24] Which is also stored/transmitted as such (see section 5.5.2), not as individually tagged measurements.

[25] Which are stored/transmitted separately from "human"/social tags.

6.3.5 Storing & sharing data

In the preferences screen of NoiseTube Mobile, shown in figure 6.7 on the next page, users can choose between three different ways to save the data[26]: submitting it to the NoiseTube community memory (CM); storing it locally on the device; or not saving it at all. The first option means that each measurement, along with (geo-)tags and timestamps, is sent[27] to the server in real-time (or with a short delay), where it is stored per track. With the second one, everything is saved to files (one per track) created on the phone's built-in memory or memory card. If the user wants, these files can later be manually uploaded to the CM using any Web browser on their phone or any PC[28]. Finally, selecting the third option ensures that no data is stored – i.e. measurements are only displayed on the screen while measuring. We had multiple motivations for this approach:

- As noted above, we wanted to design NoiseTube Mobile in a way that allows it to be used as a stand-alone personal sensing app, which does not oblige users to share anything with anyone – although of course we hope most people *will* share data.
- We wanted our users to have full access to and control over *their* own data, without first having to submit it somewhere. The locally stored track files employ a simple, human-readable XML-based format (see section D.2), allowing anyone with basic computing skills to study or use the data for whatever purpose. Moreover, track files can be shared with others without relying on the CM.
- Submitting of data to the CM had to be optional to reassure privacy-concerned users, as discussed below in section 6.3.6.2.
- The majority of mobile phone users, including many smartphone owners, still do not have access to mobile broadband[29]. While Wi-Fi hotspots can offer a (free) alternative, city-wide coverage is still rare. Although other mobile sensing researchers tend to ignore this fact, we explicitly chose not to, in order to maximise potential adoption, also in developing countries. Hence, Internet access is entirely optional while using NoiseTube Mobile. Offline usage does not stand in the way of contributing to the CM, thanks to the locally stored files that can be uploaded later via any (fixed) Internet connection.

It is noteworthy that in the market for SLM devices, "data logging" (i.e. the ability to store measurement series on the device) is a feature that commands a hefty price premium[30] (see section A.5.1.1), and network connectivity is something that is only found in the most expensive equipment. But on smartphones, which have ample built-in or extendable memory and multiple connectivity options, offering such features is fairly trivial.

[26] Note that switching the saving mode automatically ends the currently running track.

[27] Data sending happens through the Internet, either via a cellular data connection (2G/3G/4G) or Wi-Fi.

[28] After which the data is stored on the CM in the same manner as when it had been sent in real-time.

[29] Because they do not want or cannot afford to pay for it.

[30] Which is rather questionable since gigabytes worth of flash memory only cost a few Euros nowadays.

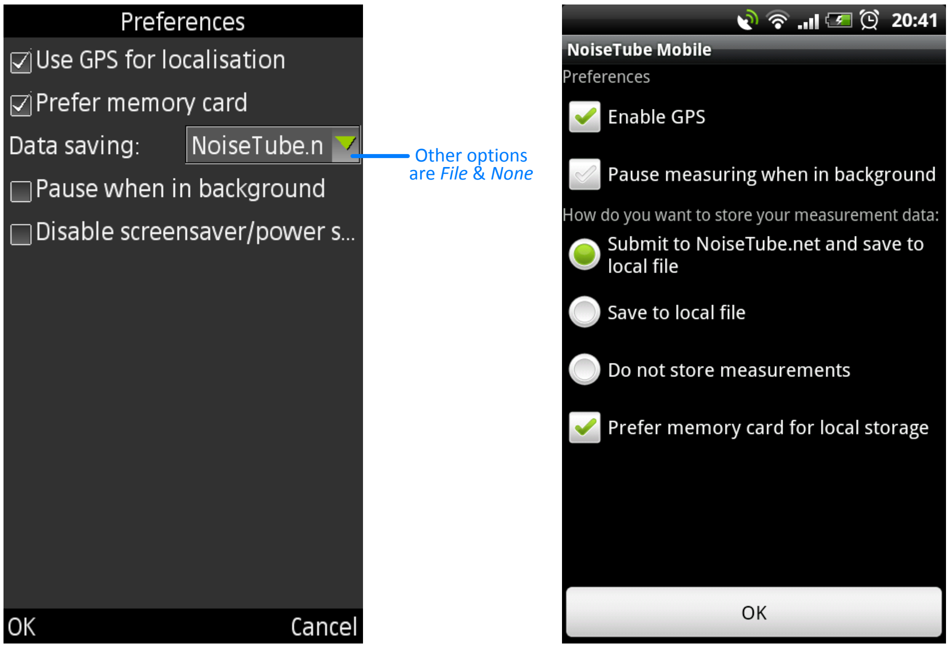


Figure 6.7: NoiseTube Mobile preferences screen
(Java ME variant on the left, Android variant on the right)

As discussed in section E.5.3, apps that rely on mobile broadband to access the Internet should be engineered to be resilient to interruptions or slowdowns of such connections. Especially if network failures are short in duration apps should deal with them autonomously without involving or informing the user. In NoiseTube Mobile, we have taken precautions to deal with this problem. Upon start-up the apps check if the NoiseTube CM server can be reached, if this fails and the user had previously chosen to submit data to the CM, then the apps automatically switch to local storage. The same check happens every time the preferences screen is opened: if the CM cannot be contacted, the option to submit data to it is disabled. To deal with network problems *while* submitting data, we use a caching mechanism: when failing to send a measurement, the apps switch from real-time submission, in which measurements are sent individually, to batch submission, in which the apps attempt to send a cached batch of measurements every 30–60 seconds. This usually leaves enough time for the connection to be restored, after which we switch back to real-time submission. All of this happens without user involvement. Even if network troubles persist – i.e. batch submission keeps on failing – no data is lost because when data is being submitted to the CM it is in fact also saved to a local track file. Later, that backup can be manually uploaded to the CM, or kept for personal reference.

6.3.6 Notes on user experience

Here we discuss some important considerations regarding the design of the user experience of the current NoiseTube Mobile apps.

6.3.6.1 Transparency & proper usage

As explained in section 3.3.4, in the context of mobile sensing "transparency" refers to the degree to which client apps interfere with other uses of the device. The more transparent the app is, the less the user is distracted, interrupted or bothered in his/her normal habits. However, unlike opportunistic sensing apps, participatory sensing apps expect users to actively decide and control where, when and for how long data is collected, which requires attention, time and effort and which may be incompatible with other activities. Hence, the latter type of sensing apps are inherently less transparent than the former.

In our concrete case, this means users should not make phone calls or type messages while measuring, because that would affect the measured sound level[31]. Moreover, when the goal is to assess the noise level at a particular place or time, users should try to avoid making loud(er) sounds themselves (e.g. by talking near the device), unless they consider those sounds to be an integral part of the studied soundscape. Finally, users should not put the device away in a pocket or bag while measuring, because that would dampen the measured sound level. For short measuring sessions the device is thus best kept in hand. For longer sessions it may be more practical to attach it to a neck cord, clip it on one's hip, or strap it to one's upper arm.

We added a number of simple features to compensate (slightly) for these inconveniences. As mentioned before, users can pause measuring at any time, for instance when they want to talk to someone or have to put their device away for some reason. Moreover, as shown in figure 6.7, both apps have an optional feature that automatically pauses measuring when they are put in the background – i.e. when the user is interacting with another app or function of the phone (e.g. to type a message). Finally, when the Android app detects an incoming or outgoing phone call, measuring is automatically paused and resumed after the call has finished. In the future we might add a similar feature to the Java ME app.

A related issue is that of autonomy. Generally speaking, the use of NoiseTube Mobile, especially when GPS in activated, can noticeably reduce autonomy. However, the degree to which this is the case varies strongly with device hardware (e.g. CPU speed, screen size), the age of the battery, GPS and cellular network reception conditions, etc. While reduced autonomy can be a concern for users – although they are free to use the app as long or as short as they want – as motivated in section 3.5.3, tackling this issue was not a priority for us. Hence, we have so far not taken steps to reduce energy consumption.

[31] On some devices even the faint sounds and slight vibrations caused by pushing keyboard buttons or touching the screen can cause noticeable sound level peaks because it happens so close to the microphone.

6.3.6.2 Privacy

Protecting user privacy is an important concern for any mobile sensing system (see challenge 6 on page 70). As opposed to fully-automated opportunistic sensing systems which may leave users with a feeling of being spied upon, NoiseTube gives them manual control over which information is disclosed. First and foremost, users decide themselves when, where and for how long measurements are being made and they can pause measuring or disable geo-tagging at any time. Secondly, users can decide on an ad-hoc (i.e. track by track) basis whether data is submitted immediately, locally stored or not stored at all. If they choose to store data locally, they can still decide to upload a selection of tracks later on, but the system does not oblige, nor coerce, them to do so in any way. Thirdly, as noted in section 5.5.1.2, even when data is submitted immediately or uploaded later on, that does not necessarily mean it is also shared with others, and users can delete it from the CM at any time. Moreover, the app only asks the user to identify him/herself with a NoiseTube account if he/she chooses to submit data to the CM.

We should also stress that the fact that NoiseTube Mobile records audio does not pose a threat to users' privacy, because it never stores, nor transmits, audio in any form. Each 1 s recording is only manipulated in RAM and is simply garbage collected after analysis. Hence there is no chance for sensitive bits of audio – e.g. of private conversations – to leak out. There are however two downsides to this approach. On the one hand, it is not possible to "re-listen" to a track, or to post (short) recordings on the CM. This might have been useful, for instance to add more tags afterwards, or to let others *hear* what the noise/sound at a certain place or time is/was like. In fact, others have created mobile sensing systems which are specifically aimed at the collection, sharing and tagging of audio fragments [35, 85, 206]. On the other, this means the computational workload to analyse audio signals is carried solely by the mobile device – i.e. it cannot be loaded off to, or shared with, the central CM server. Currently that does not pose a problem, since even smartphones from 4 years ago typically have no trouble with our real-time SPL measuring algorithm. But if we at some point would want to include additional signal processing[32] we may need to be careful – although the gigahertz multicore CPUs found in today's high-end smartphones probably provide ample headroom.

As motivated in section 3.5.3, we considered the development of advanced, cryptography-based privacy solutions to be out of scope. Hence, such solutions are not part of the current NoiseTube system. However, in a recent collaboration with cryptography specialists [135], we have tackled concerns over the disclosure of personal location traces in an entirely new way. This work, the results of which have yet to be integrated in NoiseTube Mobile and the CM, is discussed in chapter 8.

[32] For instance (semi-)automatic sound source classification.

6.3.7 Experimental features

During development we have played with some experimental features that are not included in the current publicly available NoiseTube Mobile apps. Three noteworthy examples are discussed here. More recent experimental extensions are covered in chapter 8.

6.3.7.1 Submitting data via SMS

In 2009 we briefly experimented with a fourth data saving option that submitted measurements via SMS messages, instead of through the Internet. This feature had been requested by an NGO [36] for a noise measuring campaign in Mumbai, India. Sadly the campaign was cancelled and later the SMS functionality removed from the Java ME app.

6.3.7.2 Dosemeter

A dosemeter is a special kind of SLM which is worn on the body to measure an individual's exposure to noise over a period of time and quantify the risk of hearing damage by taking both the sound level and the duration of exposure into account. As discussed in section A.5.2 and section 4.3.1, dosemeters are especially used in industrial settings. However, they could also be useful in leisure settings, for instance for concert or club goers (see section 4.2.3). It seemed interesting to evaluate if mobile phones could be used as dosemeters because most people more or less constantly carry one. Hence, we extended the Java ME app with an experimental feature that measures the *noise dose*, shown as the *D*-value in figure 6.8. This value, which is computed as advised by the American National Institute for Occupational Safety and Health, expresses cumulative noise exposure as a percentage of the maximum "safe" amount for an 8 hour working day [365]. While it works in principle, this feature is not accessible in the current Java ME app, nor in the Android app. One reason is that, because in the Java ME app SPL measuring is not truly continuous (see section 6.5.4.2), we can only guess (i.e. interpolate) the SPL in the short gaps between measurements. Another reason is that, as we will see in chapter 7, due to the properties of mobile phone microphones NoiseTube Mobile tends to be less accurate at very high levels. Because such potentially dangerous levels carry a big weight in the noise dose calculation, this can significantly reduce the accuracy of the result. However, should we come across Android phones which perform better at high levels, this functionality could be reactivated easily.

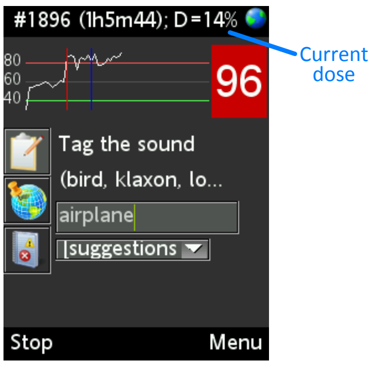


Figure 6.8: Experimental dosemeter feature in Java ME app

6.3.7.3 Frequency spectrum visualisation

To provide insight in the different frequencies occurring in measured sounds we have implemented an experimental 5-band frequency spectrum visualisation for the Android variant of NoiseTube Mobile [39: pp. 33–34]. This visualisation, shown in figure 6.9, is powered by a frequency spectrum analysis algorithm based on a *fast Fourier transform* [414: pp. 319–321, 582] implementation provided by the *KJDSS* library [197]. Ideally this visualisation would be updated every second, like the SPL is. Unfortunately the current implementation is not efficient enough to do that. Because we consider this a non-essential feature, we have so far not spent time on optimising the algorithm and have kept the feature out of the publicly released app. However, as others have recently released SLM apps for Android containing smoothly running spectrum visualisations (see section 6.6), we are confident that with the right optimisations it should be possible to make our algorithm run faster, such that this feature can be included in a future release.

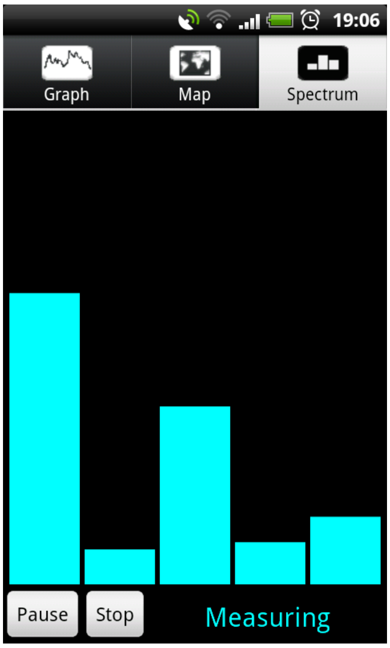


Figure 6.9: Experimental frequency spectrum visualisation in Android app

6.4 Platforms & dependencies

Researchers in computer science can often afford to develop programs for hardware and/or software platforms that are not common[33] outside (or even inside) academia, as long as it serves the research agenda and unless there are short-term commercial goals. But when researching crowdsourcing or mobile sensing systems, especially if that involves (experimental) deployments at scale, this luxury is largely absent, as platforms have to be chosen in function of the needs or tastes of potential contributors.

Our ambition to build a mobile sensing system that can be used in practice by local communities and/or individual citizens, thus requires us to pay attention to the evolution of the smartphone market. After all, there is no point in developing apps for devices that are little used by or prohibitively expensive for the intended audience. This is complicated because the smartphone market – and the definition of what constitutes a smartphone – is constantly changing, as discussed in sections E.1 and E.2. Hence, along with research goals and technical concerns, economic arguments have to be taken into account.

[33] Because they serve a particular niche or because they are expensive, experimental or even futuristic.

In this section we explain how these factors have influenced our decisions regarding targeted platforms and the timing of development efforts, which have so far resulted in the two released NoiseTube Mobile variants: the Java ME and the Android app. Moreover, we give an overview of the dependencies underpinning each app.

6.4.1 The Java ME application

When the NoiseTube project got underway in June 2008, our limited resources in terms of manpower meant that it was not sensible to target multiple mobile platforms immediately. The extra time spent programming would have slowed us down and was unlikely to result in additional research insights. Below we first explain why we chose to target the Java ME CLDC/MIDP platform, both for *Mobile Noise Tagger* (MNT), the mobile app that was part of NoiseTube Prototype (see appendix C), and for the initial NoiseTube Mobile app. Then, we give a brief overview of the dependencies of the latter app.

6.4.1.1 Why Java ME?

As discussed in section E.2.1, in mid-2008 the smartphone market – especially in Europe – was dominated by devices running Symbian OS [519], mostly from Nokia. At the time, the most popular and flexible mobile application platform was Sun's Java ME CLDC/MIDP, which we discuss in detail in section E.3. Apart from Symbian OS devices, this platform was also supported on RIM's BlackBerries as well as some smartphones from other brands.

Due to the platform's popularity and compatibility across mobile phone brands and OSs, and our experience with Java programming, we developed the MNT app for Java ME CLDC/MIDP, and tested it on a Symbian OS/S60-running Nokia N95 8GB [385][34]. As explained in appendix C, the MNT app did not make SPL measurements since that task was handled by a desktop program. When planning for the integrated solution, which became NoiseTube Mobile, we initially feared that the smartphones of the day would lack the processing power to do real-time SPL measuring, especially if implemented in Java, as opposed to a native language[35]. But when we tried it out on the Nokia N95 8GB, we were proven wrong: the device turned out to be capable of running a rudimentary real-time SPL measuring algorithm implemented in Java.

[34] Back then, the Nokia N95, and the later 8GB version, was one of the most advanced smartphones available and the device of choice of many other researchers involved in mobile sensing [293, 351, 355].

[35] For instance, apart from Java ME CLDC/MIDP apps (called MIDlets), Symbian also supports native apps written in C++ – the latter run directly on the OS, while the former run on a virtual machine.

In September 2008, with the performance concern out of the way and to benefit from the abovementioned advantages, we stuck with the Java ME CLDC/MIDP platform for NoiseTube Mobile[36], even though it was developed from scratch[37].

6.4.1.2 Dependencies

As explained in section E.3, the Java ME middleware [513] has a layered architecture. A concrete Java ME platform consists of a *configuration* layer, a *profile* layer, a layer with *optional packages* and/or *vendor APIs*, and finally the application layer at the top. Java ME CLDC/MIDP is one such platform and is aimed at mobile phones. Applications for this platform are called *MIDlets*. Hence, NoiseTube Mobile for Java ME is a MIDlet.

At the level of the device, our MIDlet/app depends on the following components:

***Connected Limited Device Configuration* (CLDC)**
: NoiseTube Mobile requires CLDC v1.1 [284] or newer; it is *not* compatible with the earlier CLDC v1.0 [279] due to limitations such as the lack of floating point support.

***Mobile Information Device Profile* (MIDP)**
: NoiseTube Mobile requires MIDP v2.0 [282] or newer. Older versions are not supported because they lack the MIDP *Media API*. Newer MIDP versions are backwards compatible so NoiseTube Mobile should work on those as well[38].

JSR-135: *Mobile Media API* (MMAPI)
: The MMAPI optional package [283] enables NoiseTube Mobile to record sound through the phone's microphone. The device must implement MMAPI v1.0 or newer, and must allow audio to be recorded in a format that can be parsed and decoded by our application (currently only raw PCM or WAVE/PCM).

JSR-179: *Location API*
: This optional package provides an API that exposes information about the physical location of the device [285]. NoiseTube Mobile requires v1.0 or newer to be supported by the device[39]. Devices that support JSR-179 commonly obtain position information from a GPS receiver (either an integrated or an external one), although the API itself is agnostic with respect to the underlying positioning technology.

[36] A secondary argument was that, as Sony CSL is after all part of the Sony Corporation, we thought – although, as far as I can remember, we were never given concrete orders in that direction – it would be good to demonstrate the app eventually on Sony Ericsson phones, many of which did not run Symbian OS but did support MIDlets.

[37] Only in 2009 some code of MNT (mostly I/O and utility classes) was carried over to NoiseTube Mobile.

[38] We know it works on MIDP v2.1, and we expect the same for MIDP v3.0, although this is untested.

[39] Obviously this was not required in the "GPS-less" derivative, which we offered until November 2010.

Apps must specify a set of criteria (in terms of accuracy, timeliness, cost, etc.) based on which a specific positioning technology is selected. NoiseTube Mobile uses sharp accuracy and timing criteria to force the use of GPS and to be able to (ideally) geo-tag each SPL measurement with new coordinates.

JSR-75: *FileConnection Optional Package* (FCOP)

The FCOP [281] specifies an API for file system access. NoiseTube Mobile requires v1.0 (or newer) of the package to be supported by the device[40]. It is mainly used to create and write to track files for local data storage, and log files for debugging.

Nokia UI API

This vendor API is found on Nokia and some Sony Ericsson phones [383, 487]. NoiseTube Mobile does not require it, but if it (v1.0 or newer) is available, the app can use it to control the phone's screensaver/power saving (see figure 6.7).

At the level of the MIDlet/app itself, we have used the following libraries[41]:

***Lightweight UI Toolkit* (LWUIT)**

We use LWUIT [516] to construct the GUI of the app. It is developed by Sun/Oracle as an open source project[42], and serves as a replacement for MIDP's outdated LCDUI toolkit – which we used for the Mobile Noise Tagger app (see appendix C).

regexp-me

The *regexp-me* library [367] is an open source[43] implementation of regular expression-based pattern matching routines, a functionality which is absent in CLDC.

Figure 6.10 summarises the stack of components NoiseTube Mobile for Java ME depends on. As a concrete example, the diagram mentions the names and/or versions of all components in the case of the app running on the Symbian-based [519] Nokia 5230 [386]. The orange parts in figure 6.10 constitute the layers of the Java ME middleware as it is implemented on this particular device by the *Java Runtime for Symbian* [380]. The green parts correspond to the MIDlets which can be installed on the phone and the additional libraries those might depend on and include. For completeness the diagram also shows native applications running alongside the Java ME runtime environment.

As explained in section E.3.4, the use of certain device features by MIDlets is restricted by means of a permissions system. To use the NoiseTube Mobile MIDlet, the user must

[40] The *PIM Optional Package*, which is also defined in JSR-75 [281], is *not* required by our MIDlet.

[41] These libraries are shipped together with the MIDlet it the same distribution package.

[42] (LWUIT is released under GPL v2.0 [194] with the "classpath exception" [584], enabling linking to non-GPL licenced programs, such as NoiseTube Mobile which is licensed under LGPL.

[43] The regexp-me library is licensed under the Apache License v2.0 [28].

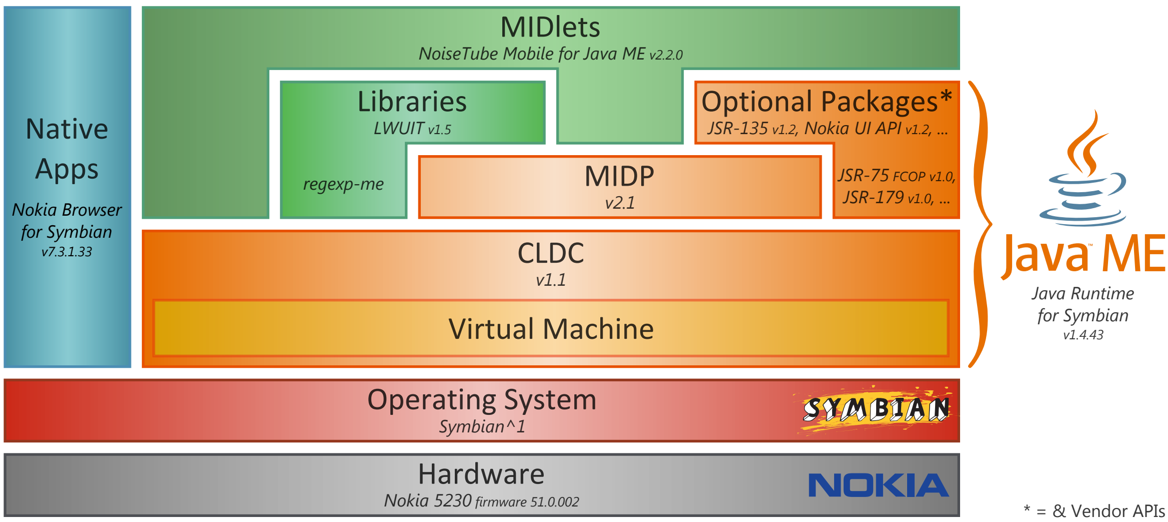


Figure 6.10: Overview of the dependencies of the Java ME variant NoiseTube Mobile, running on a Nokia 5230 phone

grant it permission to access multimedia features (i.e. to record audio), and to read and write local files. Giving permission to use positioning technology and access the Internet is optional to use the app, but the former is required for GPS-based geo-tagging, and the latter for data submission to the CM and for downloading updated calibration settings.

6.4.2 The Android application

Due to the changing market conditions and a lack of innovation in the platform and supporting devices (see section E.2.2), and various technical limitations (see section 6.5.5), we probably would not have targeted Java ME CLDC/MIDP if we would have started from scratch in late 2009, let alone today. Nevertheless, we have long resisted the temptation to port NoiseTube Mobile to one of the newer, more popular and innovative smartphone platforms, because we rather focused our attention on improving the existing system – both the CM and the mobile app – and on new research steps which did not strictly require a new app – e.g. the validation work discussed in chapter 7.

Still, the growing disadvantages of the initially chosen platform made it more and more desirable to have a NoiseTube Mobile variant for a modern platform (or multiple ones). We felt this would not only help the NoiseTube project to remain relevant in the eyes of potential adopters – individuals, communities and authorities – but also to serve as a test bed for new research. Hence in October 2010 we initiated development efforts for new platforms, with contributions from students working under our guidance. So far this has resulted in the Android app, and soon we will release one for the iOS platform.

6.4.2.1 Why Android (and iOS)?

As discussed in section E.2.2, the evolution of the smartphone market since mid-2008 clearly put Google's Android and Apple's iOS on the foreground as the most interesting new platforms for mobile sensing applications – or any mobile application for that matter. These platforms are found on both mobile phones and tablet PCs, and their app development frameworks make it easy to target both types of devices (quasi) simultaneously.

Almost form the beginning of the NoiseTube project a very common question we got from outsiders was if and when we would offer NoiseTube Mobile as an iPhone app. Consequently we have played with that idea since at least early 2009. But as iOS [30] does not support Java ME, nor any other Java flavour, this required us to rewrite the app from scratch in Objective-C (the primary language for iOS apps). Due to our limited manpower and other priorities we remained hesitant about investing time in such an effort.

Porting the Java ME app to Android seemed like lower-hanging fruit because Android apps can be written in Java, allowing existing skills and code to be reused[44]. Moreover, as discussed in section E.2.2, Android was quickly becoming the most popular smartphone platform and, more importantly, the devices come in a far wider price range than the iPhone. Another argument was that the Android app could serve as a starting point for the thesis research of one of our master students [39], involving code written in *AmbientTalk* [14, 548], which works on Android but not on CLDC, nor iOS. Last but not least, we took the development of the Android app as an opportunity to refactor the Java ME app, resulting in an improved, cross-platform architecture shared by both. For all these reasons, work on the Android app was prioritised over the iOS app.

In the process, we also took the opportunity to make several functionality and usability improvements by leveraging features of the Android platform, as shown in section 6.3. While there are alternatives (see section E.4.5), the entire Android app was programmed in Java, allowing for maximal reuse of existing, yet heavily refactored, code. We felt there was no need to use faster-running native code (written in C/C++), because the Java ME app had shown acceptable performance on devices with considerably lower processor speeds and memory sizes than the average Android device on the market in 2010[45].

6.4.2.2 Dependencies

A thorough discussion of the Android platform [226] and its components is provided in section E.4. The dependencies of our app can be summarised briefly because, compared

[44] Even though Android's class libraries and app framework are very different from Java ME CLDC/MIDP, as discussed in section 6.5.1.1 and section E.4.

[45] Still, if future extensions would require it, it is possible to write performance-critical parts in C/C++.

to Java ME CLDC/MIDP devices, the gamut of Android devices is considerably less fragmented in terms of different platform components and versions thereof.

At the level of the device, our Android app requires:

Android

On mobile phones this can be any version $\geq$ v2.1.x (codenamed *Eclair*). On tablets we require a version $\geq$ v3.0.x (codenamed *Honeycomb*)[46].

Google Maps External Library

This library [218] provides a widget that displays maps and satellite pictures served by Google Maps [216]. We use this to project measurements and tracks on top of a map, as shown in figure 6.4. While strictly speaking not a part of Android, this library is present on a large majority of Android devices in use today[47].

There are no additional dependencies at the level of the app itself[48].

Figure 6.11 summarises the stack of components the Android variant of NoiseTube Mobile depends on. As a concrete example, the diagram mentions the names and/or versions of all components in the case of the app running on a HTC Desire Z [260] smartphone with Android v2.3 *Gingerbread*.

For completeness the diagram also shows a few other apps, running alongside NoiseTube Mobile, some of which may contain native code. It is interesting to compare this diagram with that in figure 6.10, which summarises the dependencies of NoiseTube Mobile for Java ME – for an explanation of the differences and similarities, refer to section E.4.6.

As explained in section E.4.3, the usage of some device features by Android apps is restricted by means of a permissions system. Contrary to Java ME CLDC/MIDP devices, Android devices always ask the user to grant all permissions an app needs upon its installation. If the user does not agree with all permissions, the app is simply not installed. Concretely, our app requires permission to use the following restricted features: audio recording, write access to the memory card or built-in storage, querying of the phone state (to detect phone calls), Internet access, and accurate positioning (i.e. GPS).

[46] For app development purposes Android uses another versioning scheme of so-called *API levels* [223]. NoiseTube Mobile for Android requires an API level $\geq$ 7 (corresponding to v2.1.x/Eclair) on mobile phones, and $\geq$ 11 (corresponding to v3.0.x/Honeycomb) on tablets.

[47] There is no specific version to worry about because this library follows the versions of Android itself.

[48] I.e. we have not used any compiled-in libraries, like we did in the Java ME app (see section 6.4.1.2).

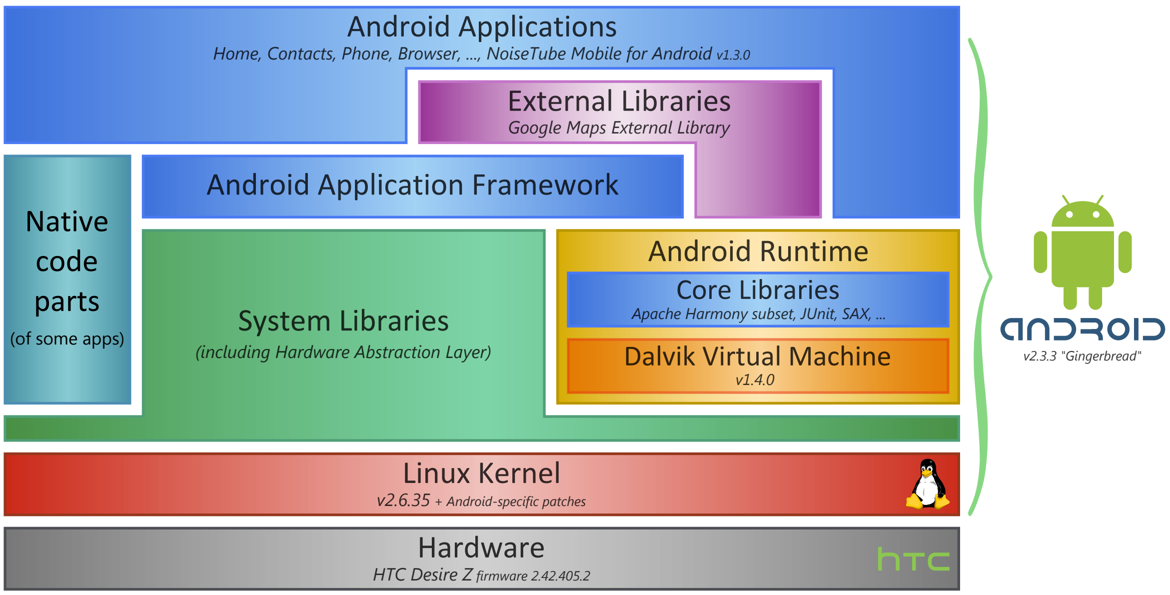


Figure 6.11: Overview of the dependencies of the Android variant of NoiseTube Mobile, running on a HTC Desire Z smartphone

6.5 Design & Implementation

In this section we discuss the design of the NoiseTube Mobile applications as well as some interesting implementation details. First we clarify the considerations that have influenced the design. Then, we discuss three principal parts of the design and implementation: the entry-point & configuration classes, the I/O classes and the measuring pipeline. Next, we look at some specific problems and solutions related to bugs or limitations of certain devices. Finally, we briefly discuss some of the ways in which (components of) the software could be reused or extended. Throughout the section, UML [191] diagrams and code snippets serve to clarify the architecture and certain implementation details.

As mentioned in section 6.2, this discussion is based on the current state of the code, including some unreleased features and refactorings. We should also note that we have made some simplifications[49] in the UML diagrams to keep them uncluttered and compact, and thereby focus the attention of the reader on the important aspects.

6.5.1 Design considerations

Before we discuss the current design and implementation of NoiseTube Mobile below, we elaborate on some important concerns that have influenced our decisions.

[49] E.g. hiding fields, methods, method arguments or return values, and sometimes even whole classes.

6.5.1.1 Cross-platform architecture

When we decided to make an Android variant of NoiseTube Mobile written in Java, it was obvious that we would try to reuse as much code from the Java ME variant as possible. Instead of just taking reusable bits and adapting them to be Android compatible, we opted for an approach in which both apps *share* part of their code. While this approach also has some disadvantages[50], we preferred it because it avoids code-duplication, which in turn allows us to at least partially maintain and improve both apps simultaneously, rather than in parallel. A more thorough motivation for this choice is discussed in [497].

However, this approach may sound easier than it is. For one thing, despite the fact that both platforms use Java, their runtime environments support different versions of the language: while the Android Runtime supports apps written in accordance with the 3rd edition of the Java language specification [229], CLDC only supports the 2nd edition [228]. For another, there is only a small set of standard classes that are available on both platforms. We refer to section E.4 for a more detailed discussion of these differences.

Consequently, the source code of NoiseTube Mobile was reorganised in three codebases: two for the platform-specific parts of each app, and a third, shared one containing a platform agnostic implementation of the main, defining behaviour of NoiseTube Mobile. In order to be platform agnostic, the shared codebase can only rely on language features and classes that are supported by both platforms. A more formal specification of what that entails exactly is given in section E.5.1.1.

To realise this cross-platform architecture and establish the shared codebase, the original Java ME codebase was refactored [192]. Wherever possible (and sensible) we extracted generic program logic and thereby separated it from Java ME-specific code. In the process we also took the opportunity to rethink several past design and implementation choices. Hence, apart from reusable code for the new Android variant, this effort also brought significant improvements to the existing Java ME variant. The first Java ME app version based on the cross-platform architecture is v2.0.0 (released in April 2011).

As explained in section E.4 and section E.5.1.1, the set of classes that are available on both platforms[51] – and can thus be relied on in the shared codebase – covers little more than the most fundamental functionalities[52], while the platforms' app development frameworks[53] are completely different. Because it only relies on fundamental Java classes, we refer to the shared codebase as a "pure Java" implementation of NoiseTube Mobile.

[50] For instance, both apps now span more classes and lines of code than separate implementations would.

[51] These classes are situated at the level of CLDC on Java ME and the Core Libraries on Android (see figures 6.10 and 6.11), the intersection of both is illustrated by figures E.4 and E.5 on page 337.

[52] Most of these classes are closely tied to the language: e.g. data types, collections, streams & exceptions.

[53] MIDP+Optional Packages on Java ME vs. Android Application Framework (see figures 6.10 and 6.11).

Of course, there is only so much we could express in pure Java. The implementation of the UI, and of more or less all core functionalities – i.e. audio recording, positioning through GPS, file system access, network communication – requires platform-specific class libraries/APIs, which cannot be dealt with directly from the shared codebase. Hence, such things had to be at least partially implemented in platform-specific code. Therefore, we let the shared codebase specify all defining behaviour on an abstract level, while leaving any concrete, platform-specific details to be filled in by the platform-specific codebases. As we will show below, we applied the *Gang of Four*'s *Abstract Factory* design pattern [198: pp. 87–95] to govern the interaction between abstract/shared and concrete/platform-specific classes. Figure 6.12 illustrates the structure of this design pattern and mentions to which codebase each class belongs.

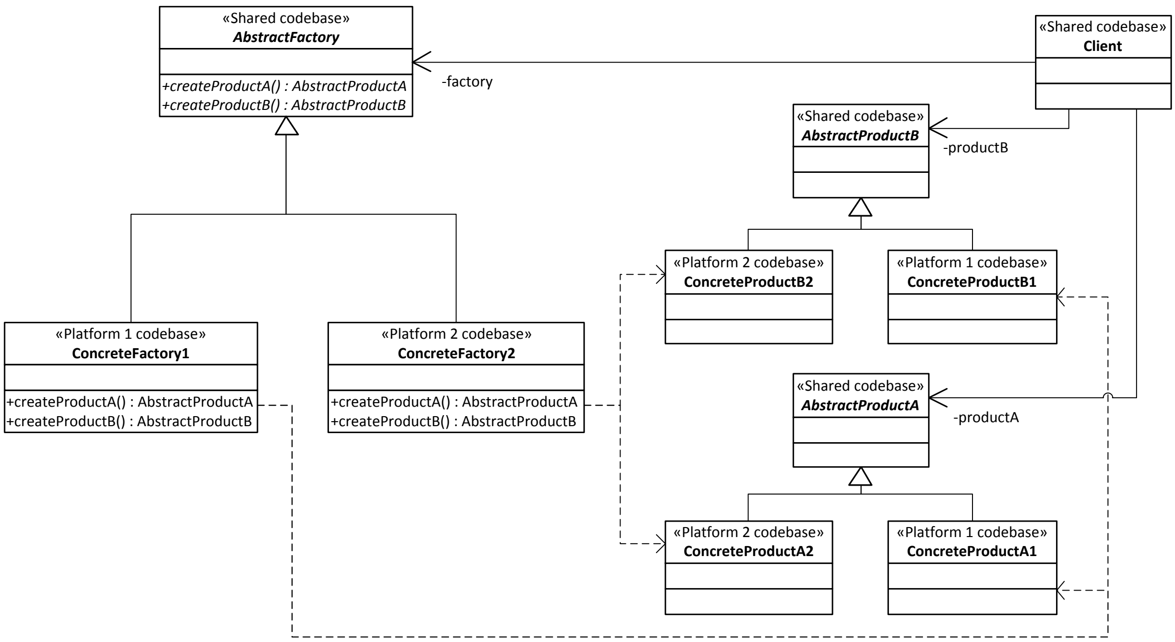


Figure 6.12: UML class diagram illustrating the *Abstract Factory* design pattern

6.5.1.2 Robustness & testability

As we explain in detail in section E.5.2, *device variability* is an underestimated challenge of mobile app development. When an app has to be deployed across a wide range of device brands and models, developers have to take certain soft- and hardware differences into account because they can undermine the *robustness* of the app. These differences may apply to the versions of the platform or its components, the concrete implementation of the platform, undocumented limitations or bugs, and hardware details.

During the development and maintenance of NoiseTube Mobile, especially regarding the Java ME variant, the need to resolve complications caused by device variability has taken up considerable amounts of time. As we show below, these complications have affected the design of the software in multiple areas. Generally speaking, our approach to make the apps robust was to check all (possibly) relevant device parameters – brand, model, platform (component) version(s), support for various APIs or features, user permissions, etc. – during the start-up procedure. This allows to detect and signal problems early – i.e. show an error message if required functionality is unavailable – and to configure or adapt the app – e.g. trigger work-arounds for known bugs (see section 6.5.5).

A related concern is *testability*. Because it is not feasible to test the apps ourselves on devices of all brands and models, we ensured that the users can help us. Therefore, both apps save all information regarding device parameters and any problems that occur in a log file. When a user reports a problem the log usually helps us to find the cause.

6.5.1.3 Dealing with audio signals

The need to record, interpret and process digital audio signals represents a specific device variability problem. In section 6.3.1 we already mentioned that, due to the fact that different mobile phone models have different microphones, our software must be calibrated for each model. However, the situation is more complex than that because, besides the microphone, there are other differences and uncertainties regarding how audio is recorded by different phones. A general introduction to audio signals is provided in section A.4. Of particular interest is figure A.6 on page 289, which summarises the stages of digital audio recording and playback. Below, figure 6.13 illustrates the situation for NoiseTube Mobile by mapping the recording stages onto the hard- and software of a mobile phone[54]:

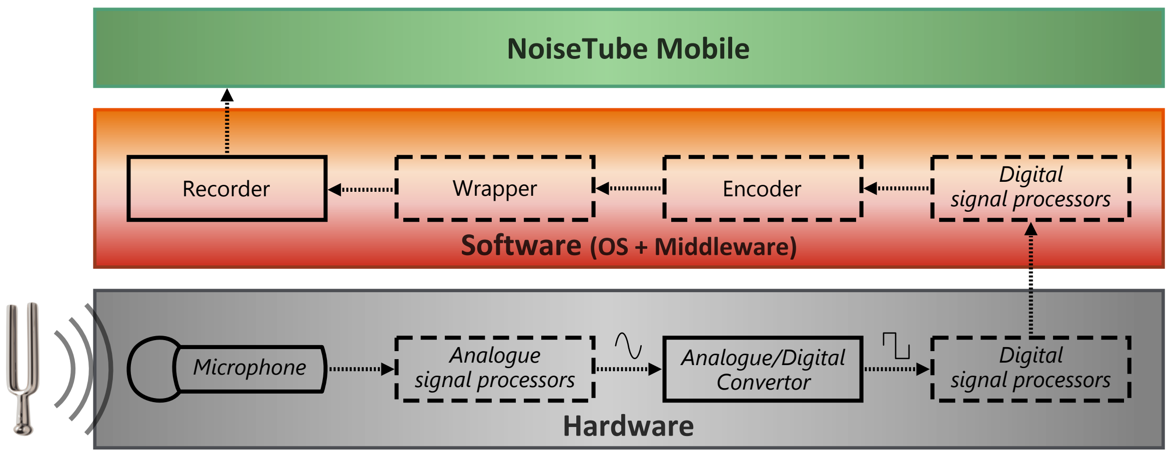


Figure 6.13: Digital audio recording path on mobile phones

In this diagram, the boxes with a dashed outline represent components which may or may not be present on a particular device, and whose presence may be undetectable from the

[54] Note that on some devices the encoder and wrapper may be implemented in hard- instead of software.

perspective of the app. Labels in *italics* signify that the parameters or properties of the component in question are unknown or uncontrollable from the perspective of the app. Here is an overview of the stages/components and the problems they can pose:

Microphone

Mobile phone manufacturers rarely (if ever) publish the specifications of the microphones they integrate in their products. Hence, although parameters such as dynamic range and frequency response (see section A.4) vary among phone models, they cannot be known upfront, nor queried programmatically.

Analogue & Digital Signal Processors

The audio signal can be filtered, amplified or otherwise processed, before (by an ASP) and/or after (by a DSP) being digitised. Whether or not this is the case, and what the parameters of any ASPs or DSPs are, is usually not documented and may be hard or impossible to determine or change programmatically. Particularly troubling for our purposes is that some phones may apply signal processing to cancel echoes or noise, or to emphasise voice frequencies.

Analogue/Digital Convertor

As explained in section A.4.2.1, the main ADC parameters are sampling rate and bit depth. To maximise fidelity to the analogue signal, and hence the original sound, both should be set as high as possible. Moreover, to correctly interpret the resulting audio stream, NoiseTube Mobile must be able to specify or at least determine these parameters. In principle, both Java ME and Android provide a way for apps to specify the sampling rate and bit depth of recorded audio. However, supported values, especially for sampling rate, tend to vary between models[55]. Moreover, we have come across models that either refuse to record audio if these parameters are specified, or that ignore what is specified and thus produce streams with unexpected properties. In both cases, the correct properties must be either determined from the stream itself, or, if that is impossible, must be hardcoded for the specific brand/-model. There are two cases in which the parameters cannot be determined from the stream. One is that the stream is "header-less" or "raw" (see below), which is common on Android. The other is that the values indicated by the stream (header) are incorrect, as we experienced on some Java ME devices from Sony Ericsson.

Encoder & Wrapper

As discussed in section A.4.2.2, there are many different digital audio formats, which can be either encodings or containers. On some phones the stream produced by the ADC, usually a PCM – the de facto uncompressed audio encoding – stream, is or can be converted by an encoder into another, usually compressed, encoding; and the raw stream, whether first encoded or not, is or can be wrapped in a

[55] Requiring a trial-and-error approach to find working settings.

container, one of the purposes of which is to describe the properties of the contained stream using a header. Hence, in order to process the recorded audio stream NoiseTube Mobile must be able to parse (i.e. “unwrap”) and possibly decode it. This implies that the app can specify – which is possible on both platforms – or determine – to handle unexpected results – the encoding and container format of the stream, and that it supports, or can be easily extended to support, various encoding and container formats, requiring specific decoding and parsing algorithms.

Recorder

The recorder component in figure 6.13 represents a class (or classes) from the phone’s application framework[56] that exposes the audio recording functionality to apps, and may or may not allow them to specify the abovementioned audio stream parameters. Because these classes are different in both platforms, we cannot control audio recording from within the shared codebase.

Our solution to deal with these problems consists of four parts:

- We have pursued a thorough separation of concerns in all dealings with audio signals. Concretely, we have spread the tasks of recording, parsing, decoding and processing audio streams across different classes. Moreover, we have relied on inheritance and polymorphism to delegate the parsing and decoding of different audio formats to separate, interchangeable subclasses.
- We have introduced a dedicated class to pre- and describe audio stream parameters. As we will show in section 6.5.4.2, this `AudioSpecification` class serves as a “contract” for audio recording: it prescribes expected stream parameters, against which the resulting stream can be checked, and it describes (raw) streams such that they can be correctly parsed, decoded and processed.
- Testing and configuring of audio recording takes place during the apps’ start-up procedure (also see section 6.5.1.2 above). This allows to signal problems early, for instance when the device refuses to record audio, or it can only record in formats that are not (yet) supported. Moreover, both apps always pick the best, working and supported configuration for the device in question, such that when SPL measuring is started, signal fidelity is maximised and the app knows what to specify/expect.
- To compensate for microphone (and possibly ASP/DSP) chacteristics NoiseTube Mobile can be calibrated for specific models. As explained in section 6.3.1, both apps come with built-in settings for a number of models and can automatically download additional ones. When there is no match for the exact model, a calibration for another model of the same brand is taken, and when there are no settings from that brand an overall default setting is used. More details, including how exactly measurements are corrected based on the calibration, are given in section 6.5.4.2.

[56] Java ME’s MIDP and MMAPI, or the Android Application Framework (see sections 6.4.1.2 and 6.4.2.2).

6.5.1.4 Reusability & extendibility

As indicated in section 3.5.2, it is our intention to contribute to the tackling of mobile sensing challenge 5 (see section 3.4) by designing the NoiseTube system with reusability and extendibility in mind, and by releasing the source code under permissive terms. In NoiseTube Mobile, the cross-platform architecture discussed above represents a major effort to enable code to be reused, rather than copied or rewritten. But what may be more relevant to outsiders is that, by adhering to software engineering principles such as *separation of concerns* [129] and *low coupling & high cohesion* [503], the current design also provides many options for extension and reuse of components in other contexts. Some concrete possibilities are sketched in section 6.5.6.

6.5.2 Entry-point & configuration classes

The class diagram in figure 6.14 shows the entry-point and configuration classes of both apps, and indicates to which codebase – shared, Java ME or Android – or underlying platform – Java ME or Android – each class belongs. This is a first illustration of how we have applied the Abstract Factory design pattern [198: pp. 87–95] (see section 6.5.1.1).

The shared `NTClient` class takes a central position in NoiseTube Mobile. First and foremost, it acts as the «AbstractFactory» in the eponymous pattern: it declares an abstract interface for the creation of «AbstractProducts», such as the configuration classes `Device` and `Preferences`, by methods `createDevice()` and `createPreferences()`. In the platform-specific codebases the role of «ConcreteFactory» is played by its subclasses `JavaMENTClient` or `AndroidNTClient`, which respectively create instances of the «ConcreteProducts» `JavaMEDevice` and `JavaMEPreferences`, or `AndroidDevice` and `AndroidPreferences`, by implementing the abstract interface.

The role of the «Client» in the pattern is played by `MainMIDlet` or `MainActivity`, respectively the entry-point of the Java ME or Android app. Immediately after being created themselves these classes create an instance of respectively `JavaMENTClient` or `AndroidNTClient`, in turn this calls the `initialize()` method of the `NTClient` superclass. This method contains the shared start-up procedure: it instantiates the configuration classes using `createDevice()` and `createPreferences()`[57], it logs device parameters and checks if all require functionalities are available, using methods of the `Device` class such as `supportsAudioRecording()` (see section 6.5.4.2).

To provide global access to a single instance of itself – via the static `getInstance()` method – `NTClient` also plays to role of the «Singleton» in the eponymous design

[57] This means `NTClient` in fact also acts as the (or its own) «Client» in the Abstract Factory pattern.

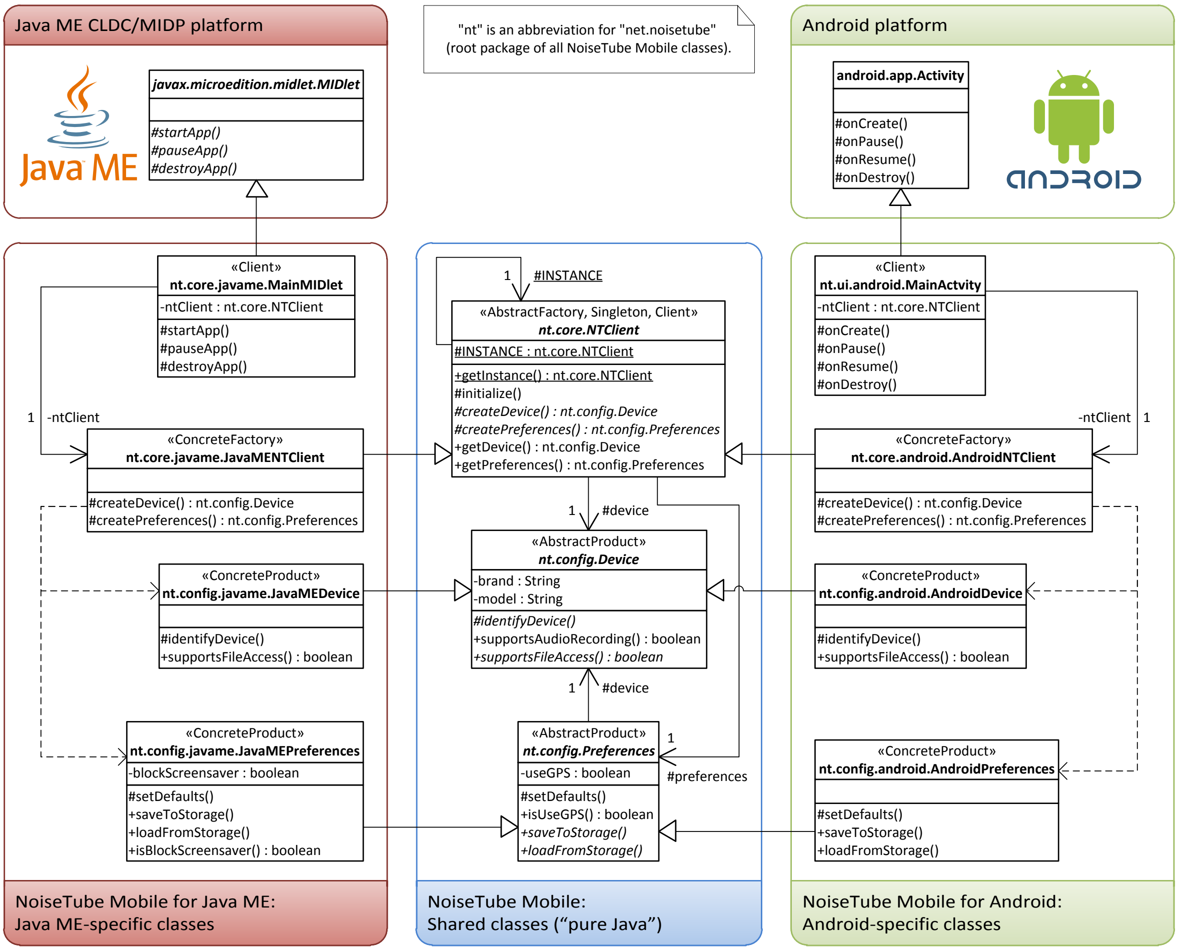


Figure 6.14: Entry-point & configuration classes

pattern [198: pp. 127–134]. Indirectly this also creates a global access point to the configuration class instances – i.e. via `getDevice()` and `getPreferences()`.

NoiseTube Mobile has two levels of configuration, the device-level, represented by the `Device`, and the user-level, represented by the `Preferences`. The former class, and its platform-specific subclasses, contains all the code that checks device parameters, which the app should take into account (see section 6.5.1.2). The latter class holds the preferences of the user – e.g. whether or not GPS can be used. The platform-specific subclasses of `Preferences` hold additional platform-specific settings – e.g. screensaver blocking on Java ME – and implement the persistent storage of all user settings.

6.5.3 Input/Output classes

The diagram in figure 6.15 shows the 3 types of I/O classes used in NoiseTube Mobile:

- The `FileWriter` class handles write access to locally stored files. It supports creation, renaming, deletion and writing of textual data. This is used for local storage of measurement in track files, logging of device parameters and debugging information, and storage of downloaded calibration settings (see section 6.5.4.2).

- The `HttpClient` class handles the communication with Web servers via the HTTP protocol. It only supports the basic HTTP request types `GET` and `POST`. This class is used to communicate with the NoiseTube Community Memory.

- The `ResourceReader` class handles the of compiled-in resource files. We use this to access the compiled-in calibration settings (see section 6.5.4.2)

Because the required APIs for these I/O functionalities are different on both platforms, each of these classes is an «AbstractProduct», subclassed by a «ConcreteProduct» for each platform. Again, `NTClient` and its subclasses play the role of respectively the «AbstractFactory» and the «ConcreteFactories».

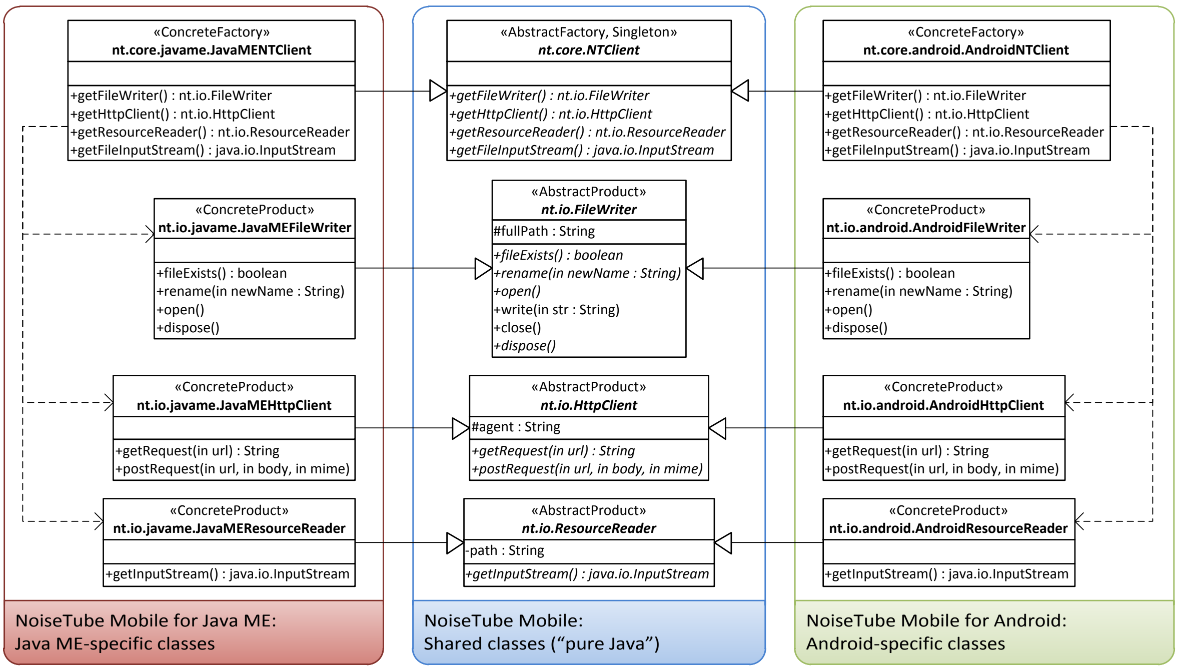


Figure 6.15: Input/Output classes

Another I/O functionality is the reading of local files. We did not introduce a dedicated "`FileReader`" class, because, once opened, files can be read with an `InputStream`, which is a fundamental Java class available on both platforms. In their implementation of the `getFileInputStream()` method the subclasses of `NTClient` use platform-specific APIs to locate and open a file and then return an `InputStream`. We use this to read previously downloaded calibration settings (see section 6.5.4.2).

6.5.4 Measuring pipeline

Now we discuss the structures and interactions that implement the main functionalities of NoiseTube Mobile. We call this the *measuring pipeline*. As discussed in section 3.3.1.1, the functionality of mobile sensing systems can usually be split in 3 stages: *sensing*, *learning* and "*closing the loop*". These same stages can also be found in the measuring pipeline, as illustrated by the sequence diagram in figure 6.16:

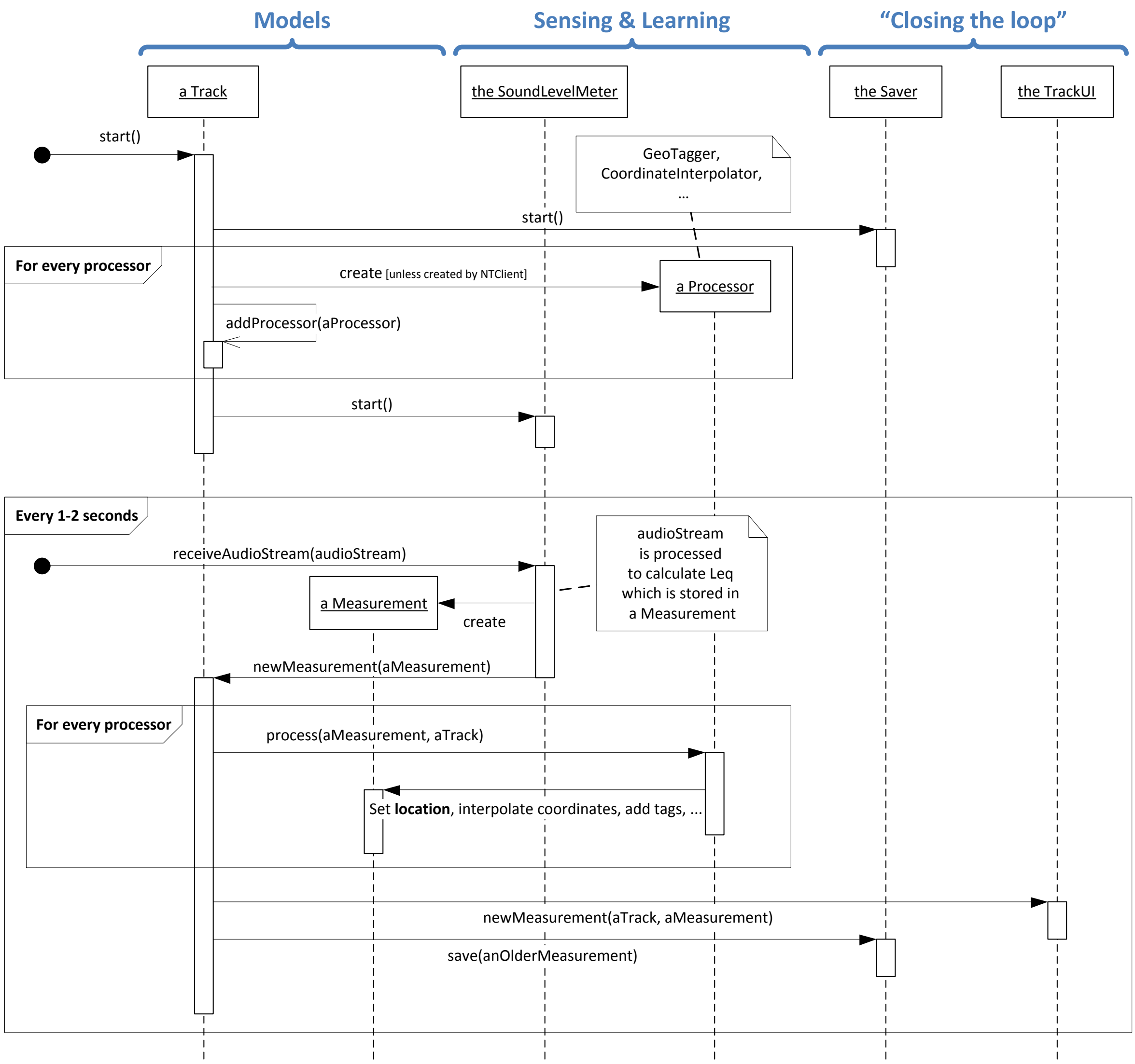


Figure 6.16: Interactions of the main objects in the measuring pipeline

6.5.4.1 Model & processing classes

The diagram in figure 6.17 shows the classes that represent data at runtime (i.e. models), and analyse or process it (i.e. processors). These are all part of the shared codebase.

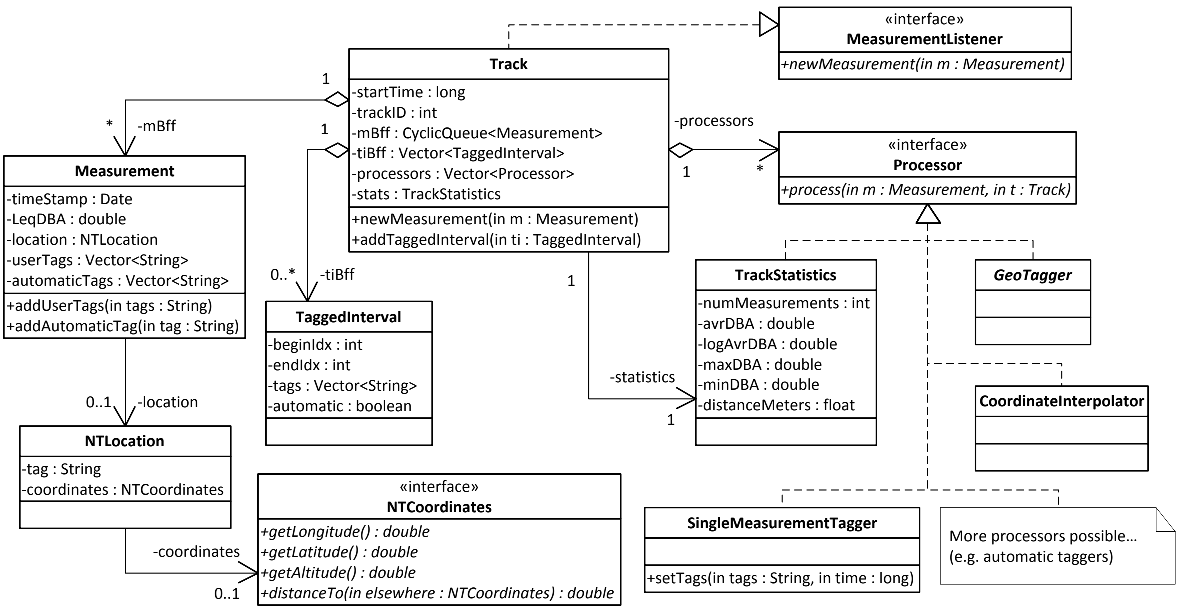


Figure 6.17: Model and processing classes

`Measurement`

Each instance of this model class holds a single SPL measurement – i.e. an $L_{\text{Aeq,1s}}$ value expressed in dB(A) – and a timestamp. If the measurement is geo-tagged its instance also holds an `NTLocation` instance (see below). If it has been (individually) tagged by the user or by a `Processor`, those tags are respectively stored in the `userTags` or `automaticTags` `Vectors`. An instance of `Measurement` is thus a local representation of a *measurement* record, with associated *tagging* records, as stored in the NoiseTube CM (see section 5.5.2).

`NTLocation` & `NTCoordinates`

An instance of the `NTLocation` class describes a location, either by a "location tag", an instance of `NTCoordinates`, or both. Geo-tagging by means of user-typed location tags is only exposed in the UI of the Java ME app (see section 6.3.2). The `NTCoordinates` interface declares getters for long-, lat- and altitude fields and methods to compute the distance and course to another `NTCoordinates` instance. The interface is implemented by `JavaMENTCoordinates` and `Android-NTCoordinates` (see figure 6.21). This extra abstraction is needed due to the incompatible coordinates/location representations of both platforms.

`TaggedInterval`

This model represents a tagged interval of measurements of a track: the interval is delimited by means of two indexes, tags are stored in a `Vector` and the type of tagging (user/social or automatic) is stored in a `boolean`. An instance of this model class is thus a local representation of a *taggedinterval* record, with associated *tagging* records, as stored in the NoiseTube CM (see section 5.5.2).

`Track` & `Processors`

An instance of the `Track` model represents a measuring session, and is thus a local representation of a *track* record as stored in the NoiseTube CM (see section 5.5.2). `Track` implements the `MeasurementListener` interface in order to take part in a *publish-subscribe* interaction with `SoundLevelMeter`, from which it regularly receives new `Measurement` instances, as shown in figure 6.16[58]. A `Track` instance holds a collection of `Processor` instances. The `Processor` interface is implemented by any class that needs to analyse, modify, tag or otherwise process measurements. This can fall under either the *sensing* or the *learning* stage. As shown in figure 6.16, whenever a new measurement is received the `Track` calls the `process()` method of each of its `Processors`, passing the new measurement as wells as itself as arguments. A `Track` also holds a fixed-size buffer with up to 60 `Measurement` instances. When a new measurement is received and the buffer is full, the oldest measurement is removed and saved to file or transmitted to the CM, to make room for the new one. The buffer is thus a sliding window of measurements that have yet to be saved/transmitted. This serves two purposes: on the one hand it is used to draw the SPL graph in the UI (see figure 6.2), on the other it allows the `Processors` to consider (and possibly modify) the recent history of measurements, rather than only the last one. Although new `TaggedIntervals` are immediately saved/transmitted they are also kept in a buffer in order to highlight them in the graph (see figure 6.6). When the last measurement to which an interval applies is saved/transmitted the corresponding `TaggedInterval` instance is removed from the buffer.

`TrackStatistics`

This class implements the `Processor` interface and is used to keep statistics about the on-going measuring session. Every time it receives a new measurement, via the `process()` method, the `TrackStatistics` updates these statistics:

- the number of measurements made so far;
- the maximum and minimum $L_{\text{Aeq,1s}}$ value measured so far;
- the arithmetic mean (i.e. simple average) of all $L_{\text{Aeq,1s}}$ values measured so far;
- the logarithmic composite average or overall $L_{\text{Aeq},T}$[59], computed from all $L_{\text{Aeq,1s}}$ values measured so far, as specified by equation A.27 on page 303;
- the total distance travelled so far[60].

This information is used in the UI of both apps, as shown in figures 6.2 and 6.3.

[58] This interaction is similar to the *Observer* design pattern [198: pp. 293–303]: `Track` acts as the *subscriber* or «ConcreteObserver», `MeasurementListener` is an «(Abstract)Observer», and `SoundLevelMeter` is the *publisher* or «(Concrete)Subject».

[59] In which T is current the duration of the track.

[60] Only if the measurement is geo-tagged with GPS coordinates.

6.5.4.2 Sound level measuring

NoiseTube Mobile's main sensing functionality is of course the measuring of the ambient sound pressure level (SPL). In section 6.3.1 we described this functionality from the perspective of the user, here we discuss how it is implemented. This code is also used in our *NoiseTube Phone Tester* apps, which we use to determine calibration settings.

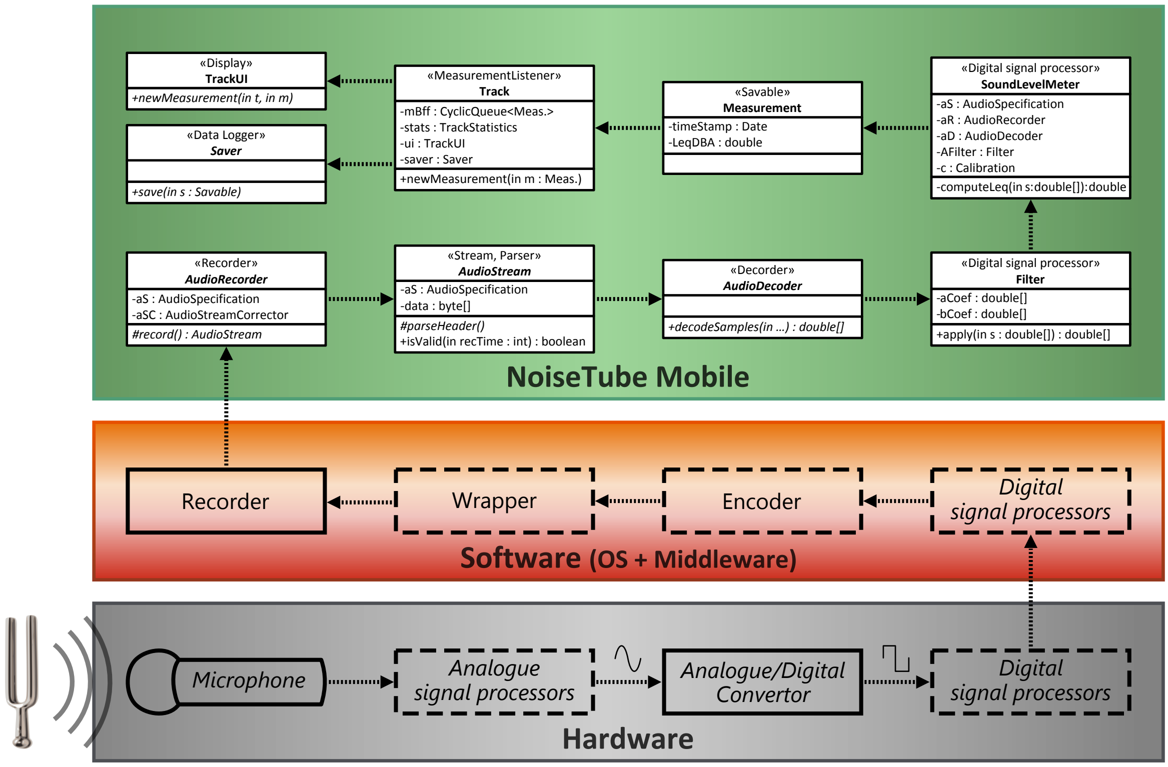


Figure 6.18: Chain of device hard- and software components and NoiseTube Mobile classes involved in sound level measuring

The main involved classes are shown in figure 6.18. Rather than to use a conventional UML class or sequence diagram, we have opted to emphasise the task each class fulfils in recording, parsing and decoding of digital audio signals (see section 6.5.1.3) and in processing them to measure SPL like a SLM (see section A.5.1). For proper understanding, it may help to compare this diagram with those in figures A.6 and A.7 on pages 289 and 290, which respectively illustrate the stages of digital audio recording/playback, and the components of a typical SLM. The hard- and software components in the lower half, and most classes[61] in the upper half of figure 6.18, fulfil the same task as a component in figure A.6 or A.7. All classes shown in figure 6.18 are part of the shared codebase, but, since the audio recording APIs of Java ME and Android are different, some of them have platform-specific subclasses. As shown in figure 6.19 we have again applied the Abstract Factory design pattern [198: pp. 87–95] to instantiate these subclasses.

[61] As indicated by the «*stereotypes*».

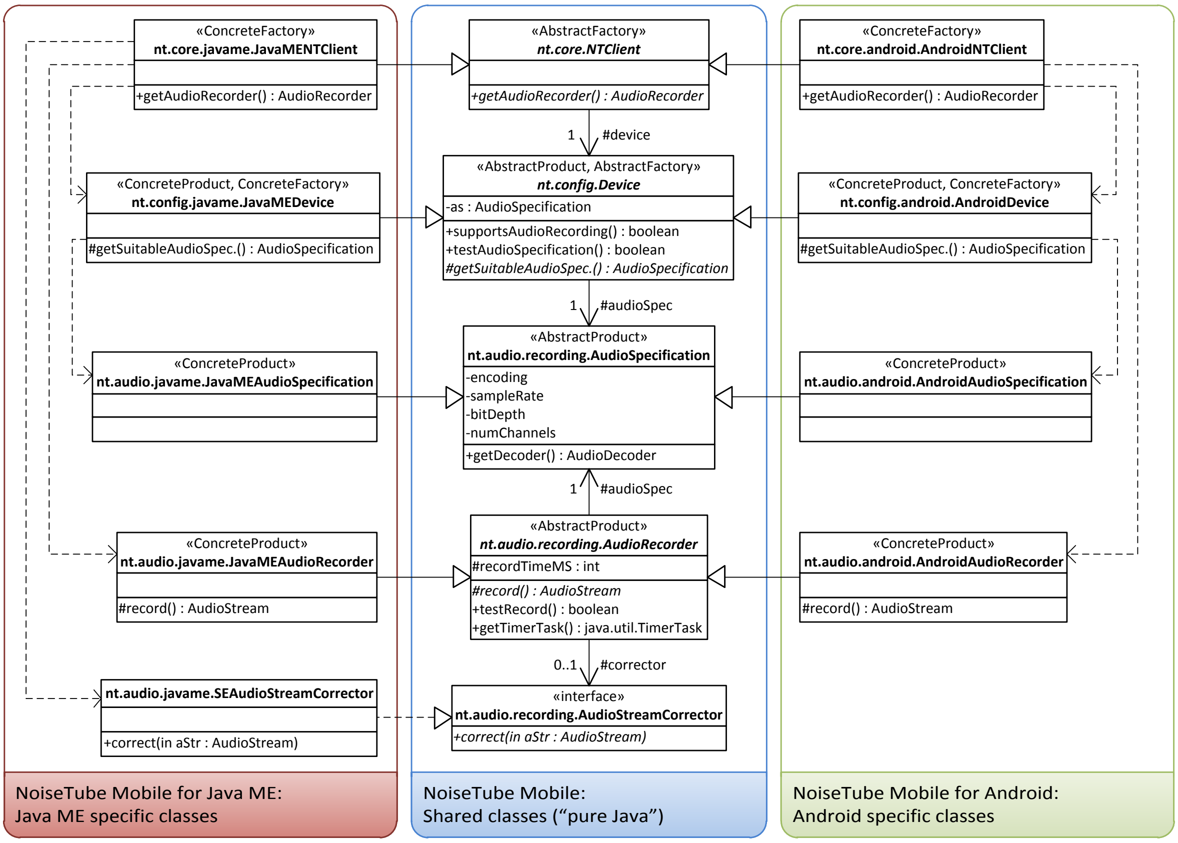


Figure 6.19: Shared and platform-specific classes involved in audio recording

Now we take a more detailed look at each of the classes involved in sound level measuring:

`AudioSpecification` & testing procedure

As motivated in section 6.5.1.3, this class serves to pre- and describe audio stream parameters such as sampling rate, bit depth, channel count and encoding. The subclasses, `JavaME`- and `AndroidAudioSpecification`, handle the details of how audio stream parameters are represented on both platforms.

`AudioSpecification` plays a central role in the testing of audio recording capabilities during app start-up. The testing procedure starts when `NTClient` calls the `supportsAudioRecording()` method of `Device`, which in turn calls its `getSuitableAudioSpecification()` method, provided by a platform-specific subclass. Here the phone's audio recording capabilities (e.g. supported encodings) are queried, and a bunch of `AudioSpecification` instances are created, sorted by desirability (e.g. favouring higher over lower sampling rates), and tried out in that order. To try out an `AudioSpecification` it is passed to the `testAudioSpecification()` method, which requests a platform-specific `AudioRecorder` instance, configured with the specification, from `NTClient` and then calls `testRecord()` on it (see below), returning the `boolean` result back to `getSuitableAudioSpecification()`. If the result is `true` the `Audio-`

Specification in question is returned and remembered by the Device as the one to use for SPL measuring (see below), if not the next AudioSpecification is tried. If no suitable specification is found supportsAudioRecording() will return false and the user will see an error message.

AudioRecorder

This class is responsible for recording bits of audio, in the format specified by an AudioSpecification instance, and of a duration indicated by recordTimeMS (for SPL measuring this is 1000 ms, for testing shorter times can be used). The abstract record() method is implemented by subclasses JavaME- and Android-AudioRecorder using platform-specific APIs. Here the actual recording is done and the data is packaged in a suitable AudioStream instance (see below) and then returned. Due to platform-specific limitations the record() implementation of JavaMEAudioRecorder starts and stops recording every time, leaving a gap of up to 1 s during which nothing is recorded (and thus no SPL measured). In contrast, AndroidAudioRecorder can record continuously to a buffer, from which record() reads a segment of 1 s every time it is called. The testRecord() method makes a short recording using record() and then checks whether resulting the AudioStream is valid with respect to the specification, has the expected duration, and whether a suitable AudioDecorder (see below) is available, if all conditions are met it returns true. During SPL measuring AudioRecorder works asynchronously. To start measuring the SoundLevelMeter (see below) calls the getTimerTask() method and schedules the returned TimerTask to run every second on a separate tread. This task executes the record() method and sends the AudioStream instance to the SoundLevelMeter.

AudioStream

This class acts both as a stream and a parser for it. Each instance holds a byte array with the data of an audio stream, and an AudioSpecification that was used to record it. The AudioStream class is abstract and must be subclassed to add parsing support for different audio container formats. Currently two subclasses are available, WAVEAudioStream and RawAudioStream. The former implements support for the popular WAVE container format [595], which is commonly used on Java ME devices. Here the parseHeader() method checks if the stream is conformant to the WAVE format and reads the properties of the contained stream (e.g. sampling rate, bit depth, etc.) such that they can be checked with respect to the specification. The latter subclass is used to hold raw audio streams, which are common on Android. These do not need to be parsed – hence parseHeader() does nothing meaningful here – just decoded. Because a raw stream does not describe itself the AudioSpecification cannot be used to validate its properties, and thus only serves to describe the (expected) properties such that the stream can be decoded and processed.

`AudioStreamCorrector`

This interface is to be implemented by classes that apply corrections to an `AudioStream`. If it has been configured with an `AudioStreamCorrector` instance the `AudioRecorder` passes every recorded `AudioStream` to the `correct()` method, before it is checked for validity and/or sent to `SoundLevelMeter`. This was introduced to resolve a problem on some Java ME devices from Sony Ericsson (e.g. the W995 [486]). On these phones the WAVE header reports a sampling rate of 8 kHz while the actual contained stream is sampled at 16 kHz. Therefore, the `SEAudioStreamCorrector` class, which implements the interface and is only used when running on an affected device, corrects the header.

`AudioDecoder`

The role of this class is to decode `AudioStream`s. It foresees methods to decode a single or a series of samples (i.e. amplitude values) to either a double precision integer or double precision floating point[62] number or an array thereof. The class is abstract and must be subclassed to add decoding support for different audio encoding formats. Currently the only supported encoding is (L)PCM [588] (see section A.4.2.1) which is decoded by the `PCMAudioDecoder` subclass. While a large majority of Java ME devices (e.g. those of Nokia and Sony Ericsson) are able to record audio in PCM, some others (e.g. those from Samsung) can only record in other, compressed formats such as AMR [574], which are not supported. Although the current architecture makes it easy to add decoders for such formats[63], it is unlikely that we will invest time in that because most (if not all) Java ME devices on sale today[64], and all Android devices, can record PCM-encoded audio.

`Filter`

This class contains a digital filter [262: pp. 317–341] implementation which we use to apply frequency weighting to the audio signal (see section A.5.1.2). As motivated in section 6.3.1, we apply A-weighting in order to get SPL measurements in dB(A). Currently the `Filter` class contains 6th order coefficients to apply A-weighting to signals sampled at 8, 16, 22.05, 24, 32, 44.1 or 48 kHz – which covers all common rates used for audio signals on mobile phones. We determined these coefficients using the *Octave toolbox for MATLAB* by Couvreur [103]. As an example, tables 6.2a and 6.2b respectively list the coefficients for sampling rates of 16 and 48 kHz.

Because the `Filter` class is implemented as a generic, digital *infinite impulse response* (IIR) filter [262: pp. 454–578], adding support for other frequency-weightings (e.g. C-weighting) and/or other sampling rates would just be a matter of adding the right coefficients. Box 6.1 shows the main part of the implementation.

[62] With a value from $[-1.0, +1.0]$.

[63] Whether the decoding algorithm is documented and easy to implement is of course another matter.

[64] For instance, Samsung no longer makes Java ME devices.

Index	*a*-coefficients	*b*-coefficients
0	1.0	0.53148982982355708
1	-2.867832572992166100	-1.0629796596471122
2	2.221144410202319500	-0.53148982982356319
3	0.455268334788656860	2.1259593192942332
4	-0.983386863616282910	-0.53148982982355686
5	0.055929941424134225	-1.0629796596471166
6	0.118878103828561270	0.53148982982355797

(a) $f_s = 16\,\text{kHz}$

Index	*a*-coefficients	*b*-coefficients
0	1.0	0.2343017922995132
1	-4.113043408775872	-0.4686035845990264
2	6.5531217526550503	-0.23430179229951431
3	-4.9908492941633842	0.9372071691980528
4	1.7857373029375754	-0.23430179229951364
5	-0.24619059531948789	-0.46860358459902524
6	0.011224250033231334	0.23430179229951273

(b) $f_s = 48\,\text{kHz}$

Table 6.2: A-weighting coefficients for different sampling rates (f_s)

As discussed in section A.5.1.2, the requirements for A-weighting filters are specified in national and international norms for SLMs. The current international SLM standard is IEC 61672-1:2002 [268]. For selected frequencies, from 10 Hz to 20 kHz, this document lists the weighting (a positive or negative dB offset) along with a tolerance limit for Class 1 and Class 2 SLMs [268: p. 16]. To evaluate the performance of our filter we have compared it with these requirements. We did this by feeding artificially generated pure tone signals to the filter and comparing the applied weightings with those specified by the standard. The result is shown on chart 6.1. The chart shows that, at a sampling rate of 48 kHz, our A-weighting filter meets the tolerance limits for Class 1 SLMs at all frequencies except those between 10 and 12.5 Hz. We should stress that this test did not involve audible sounds captured with a microphone. Instead, the signals were generated in software. Hence, here we only tested the performance of the A-weighting filter itself. In chapter 7 we discuss the evaluation of the accuracy of NoiseTube Mobile as a whole.

```java
public double[] apply(double input[])
{
   double[] output = new double[input.length];
   for(int i = 0; i < input.length; i++)
   {
      //take sample:
      double x_i = input[i];
      //filter it:
      double y_i = x_i * bCoef[0] + conds[0];
      //store filtered sample:
      output[i] = y_i;
      //adjust all but the last condition:
      for(int j = 0; j < order - 1; j++)
         conds[j] = x_i * bCoef[j + 1] - y_i * aCoef[j + 1] + conds[j + 1];
      //adjust last condition:
      conds[order - 1] = x_i * bCoef[order] - y_i * aCoef[order];
   }
   if(!keepConds) //conditions may be kept for continuous signals
      conds = new double[order]; //clear conditions
   return output;
}
```

Box 6.1: The `apply()` method of the `Filter` class, used to apply frequency weighting

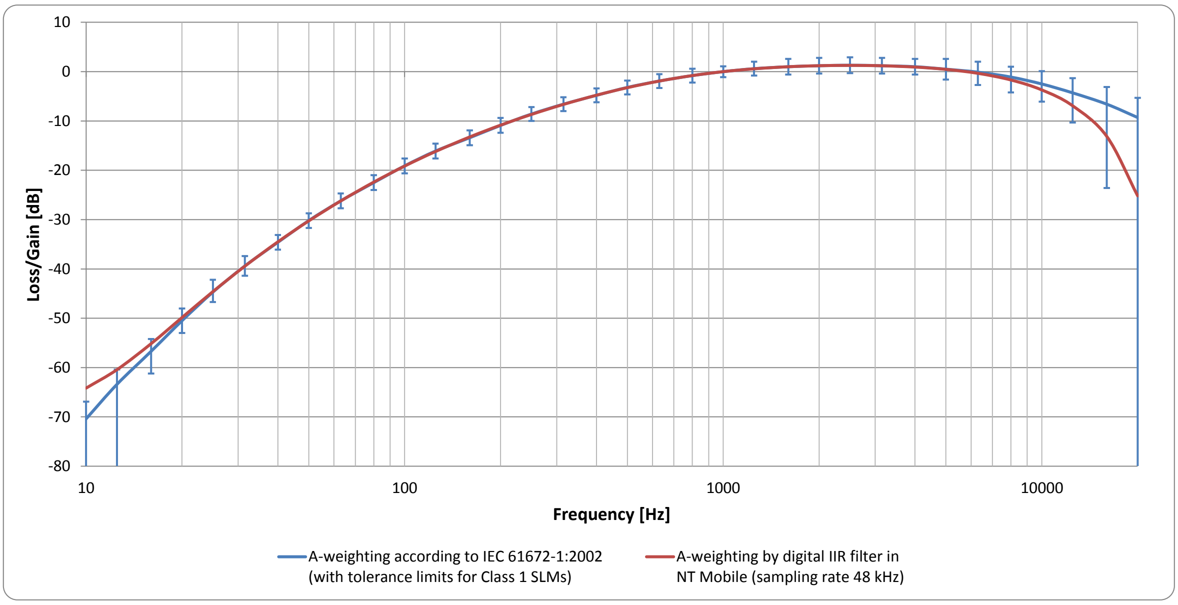


Chart 6.1: Comparison of the A-weighting filter in NoiseTube Mobile with the specification according to IEC 61672-1:2002 [268: p. 16][65]

`SoundLevelMeter`

As its name suggest this class works like an SLM: it processes audio signals to calculate SPL values. When instantiated it receives an `AudioSpecification` (i.e. the one that was selected during testing), possibly a `Calibration` (see below), and a `MeasurementListener`[66]. When the `start()` method of `SoundLevelMeter` is called it requests a platform-specific `AudioRecorder` instance from `NTClient` (configured with the `AudioSpecification`), asks it for a `TimerTask` and schedules that to run every second. From then on the `AudioRecorder` regularly calls `SoundLevelMeter`'s `receiveAudioStream()` method to pass newly recorded `AudioStreams`. For every received stream, `SoundLevelMeter` uses the decoder, which is gets by calling the `getDecoder()` method of `AudioSpecification`, to decode as many double precision floating point samples as the sampling rate in Hz (denoted f_s), representing exactly 1 s of sound. The samples are then processed by a `Filter` instance with A-weighting coefficients for the right sampling rate. Next, the A-weighted samples are passed to the `computeLeq()` method, shown in box 6.2, to compute the $L_{\text{Aeq,1s}}$. That value is then corrected with the `Calibration` (if there is one) and stored in new `Measurement` instance, to be passed to the `MeasurementListener`, as shown in figure 6.16.

As formulated by equation A.24 on page 301, calculating $L_{X\text{eq}\,T}$ involves an integration step, which, in case of a digital audio signal (composed of discrete samples), can be approximated with a Riemann sum, as shown by equation A.25 on page 302.

[65] Note that the negative tolerance limit for Class 1 SLMs at 10, 12.5 and 20,000 Hz is $-\infty$ [268: p. 16].

[66] In NoiseTube Mobile the listener is a `Track`, but in the Phone Tester apps it is a `Calibrator`, as shown in figure 6.20, which explains the need for the `MeasurementListener` interface.

```java
private double computeLeq(double samples[]) throws Exception
{  //samples.length = sampling rate (samples contains exactly 1s of audio)
   double sumsquare = 0.0d, leq;
   for(int i = 0; i < samples.length; i++)
      sumsquare += samples[i] * samples[i];
   leq = (10.0d * MathNT.log10(sumsquare / samples.length)) +
       93.97940008672037609572522210551d;
   if(Double.isNaN(leq) || leq <= 0)
      throw new IllegalResultException("Leq is NaN, negative or zero: " +
           Double.toString(leq));
   return leq;
}
```

Box 6.2: Calculation of the $L_{\mathrm{Aeq,1s}}$ by SoundLevelMeter

For $L_{\mathrm{Aeq,1s}}$ (i.e. $X = \mathrm{A}$ and $T = 1\,\mathrm{s}$), equation A.25 becomes equation 6.1, and by filling in the value of p_0 that can be reduced to equation 6.2, which is exactly how the computeLeq() method shown in box 6.2 works.

$$L_{\mathrm{Aeq,1s}} = 20 \cdot \log_{10}\left(\frac{\sqrt{\frac{1}{f_s}\sum_{i=1}^{f_s} p_{\mathrm{A}}^2(i)}}{p_0}\right) \tag{6.1}$$

$$= 10 \cdot \log_{10}\left(\frac{\frac{1}{f_s}\sum_{i=1}^{f_s} p_{\mathrm{A}}^2(i)}{p_0^2}\right)$$

$$= 10 \cdot \log_{10}\left(\frac{1}{f_s}\sum_{i=1}^{f_s} p_{\mathrm{A}}^2(i)\right) - 10 \cdot \log_{10}\left(p_0^2\right) \quad \text{With:} p_0 = 0.00002\,\mathrm{Pa}$$

$$= 10 \cdot \log_{10}\left(\frac{1}{f_s}\sum_{i=1}^{f_s} p_{\mathrm{A}}^2(i)\right) + 93.9794\ldots \quad [\mathrm{dB(A)}] \tag{6.2}$$

As discussed in section A.2.3, the p_0 value was chosen as the approximate threshold of human hearing at 1000 Hz. In a sense, the +93.9794... in computeLeq() is unimportant because the calculated values still have to be adjusted to account for the characteristics of the microphone, which happens in the Calibration class discussed below. Still, the addition ensures the values are positive and more or less in the expected range for decibels. Although we could have used another offset (e.g. +100), for aesthetic reasons we have implemented the calculation entirely as defined by acoustical conventions and standards [268], with the offset based on p_0. While negative SPL values are possible[67], computeLeq() throws an exception if the result is $\leq 0\,\mathrm{dB(A)}$. This is because practical experience has shown that such values are only computed if the AudioStream data is corrupt due to a device malfunction. After catching 4 such exceptions SoundLevelMeter automatically stops and restarts the AudioRecorder, which usually solves the problem.

[67] I.e. for very quiet sounds, usually inaudible to humans (see section A.2.3).

`Calibration` & `CalibrationFactory`

To correct SPL measurements to account for the properties of the microphone, and possibly other components of the phone's audio recording path (see section 6.5.1.3) we have introduced the `Calibration` class. The diagram in figure 6.20 shows this and related classes such as `CalibrationFactory`.

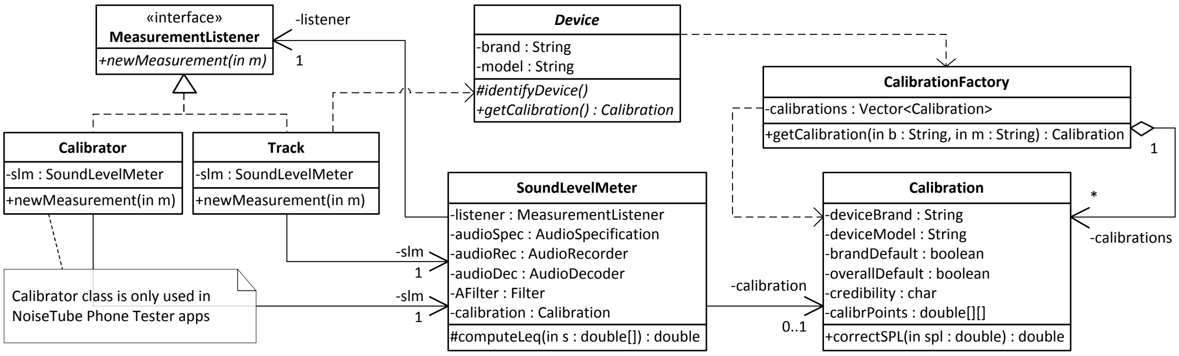


Figure 6.20: `Calibration`, `CalibrationFactory`, and related classes

A `Calibration` instance holds an array of *calibration points* specific to a certain phone, identified by its brand and model. These points are to be determined experimentally in a process that is explained, motivated and evaluated in chapter 7. For now it suffices to say that each calibration point is a pair of SPL values: one that was measured on the phone in question by `SoundLevelMeter` without any correction being applied[68]; and another that was measured, at the same time and at the same distance from the sound source, by a trusted reference device[69]. Ideally this process is repeated with sounds of increasing SPL (e.g. from 30 to 100 dB(A) in steps of 5 dB) such that we get a series of calibration points, specific to the phone. Chart 6.2 shows the calibration points for a Nokia 5230 [386] (a Java ME device) and an HTC Hero [587] (an Android device).

For a given calibration point, we call the value that was measured on the phone the *input* and the one that was measured by the reference device the *output*. As illustrated by the chart, the points give us a level-dependent indication of the (systematic) measuring error made on a particular phone with respect to the reference. This information allows us to correct measurements by adding or subtracting an offset to compensate for the error. For instance, if we measure 60 dB(A) on a Nokia 5230 then the calibration point (60.0, 65.0) tells us we should add 5 dB. This is the principal idea behind our SPL correction algorithm, implemented in the `correctSPL()` method of `Calibration`. However, because in general the to-be-corrected SPL value does not exactly match one the calibration points,

[68] This is done with the NoiseTube Phone Tester apps, whose `Calibrator` class (see figure 6.20) uses a `SoundLevelMeter` instance but does not configure it with a `Calibration`.

[69] I.e. a properly calibrated (semi-)professional SLM, which – for the sake of the calibration of the phone – is assumed to produce correct measurements.

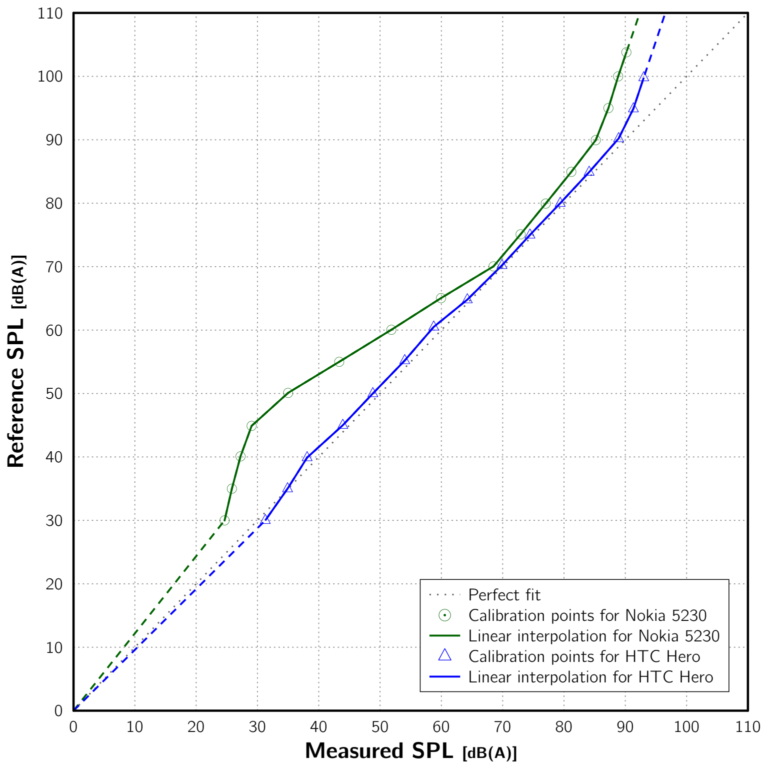


Chart 6.2: Linear interpolation between calibration points for a Nokia 5320 & an HTC Hero

we calculate the corrected value by linearly interpolating between the closest two. Graphically this is equivalent to connecting consecutive calibration points with a straight line, as shown by the full lines on chart 6.2. The dashed lines in the chart illustrate our treatment of borderline cases: if the uncorrected SPL is smaller than the input value of the first calibration point (and hence of all others), then we interpolate between the origin – where 0 dB(A) stays 0 dB(A) – and the first point; if it is larger than the input value of all points, then we interpolate between the last two points. A more formal description of the algorithm is given in box 6.3 and its implementation by the `correctSPL()` method is shown in box 6.4.

Note that our SPL correction algorithm only applies a level-dependent adjustment, not a frequency-dependent one. In other words, our approach to calibration and correction relies on the assumption that we can achieve sufficiently accurate results without considering the frequency domain. As we will discuss in chapter 7, this choice was made after careful consideration and is based on the specific type of sound (i.e. *white noise*) used to determine calibration points. A second assumption, again supported by experiments discussed in chapter 7, is that the characteristics of the microphone of one instance of phone model are sufficiently similar to those of another instance of the same model, in order to use the calibration points determined on the former to correct measurements made on the latter. In other words, if someone takes the effort to calibrate NoiseTube Mobile for a particular model, all owners of such phones can benefit from that, which is essential for scalability.

We have an array of n calibration points, specific to the phone or its brand/model, which are sorted by increasing input values:

$$C_{\text{phone}} = \left\{ \left(c_0^{\text{IN}}, c_0^{\text{OUT}}\right); \ldots; \left(c_{(n-1)}^{\text{IN}}, c_{(n-1)}^{\text{OUT}}\right) \right\} \quad \text{with } \forall i \in [0, (n-1)] : c_{(i-1)}^{\text{IN}} < c_i^{\text{IN}}$$

For every $L_{p,\text{phone}}$ (an uncorrected SPL measurement) find the index of the first calibration point whose input is bigger or equal, take the last index if $L_{p,\text{phone}}$ is bigger than all inputs:

$$i = \begin{cases} 0 & \text{if } L_{p,\text{phone}} \leq c_0^{\text{IN}} \\ j & \text{if } c_{(j-1)}^{\text{IN}} < L_{p,\text{phone}} \leq c_j^{\text{IN}} \\ n-1 & \text{if } L_{p,\text{phone}} > c_{(n-1)}^{\text{IN}} \end{cases}$$

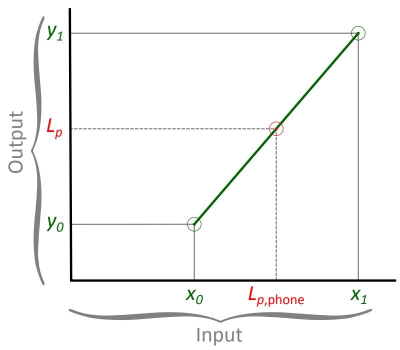
Determine the coordinates of the calibration points between which we will interpolate:

$$x_0 = \begin{cases} 0.0\,\text{dB(A)} & \text{if } i = 0 \\ c_{(i-1)}^{\text{IN}} & \text{if } i > 0 \end{cases}$$

$$y_0 = \begin{cases} 0.0\,\text{dB(A)} & \text{if } i = 0 \\ c_{(i-1)}^{\text{OUT}} & \text{if } i > 0 \end{cases}$$

$$x_1 = c_i^{\text{IN}}$$

$$y_1 = c_i^{\text{OUT}}$$



Apply linear interpolation to compute the corrected L_p value:

$$L_p = \frac{(L_{p,\text{phone}} - x_0) \cdot (y_1 - y_0)}{(x_1 - x_0)} + y_0 \tag{6.3}$$

Box 6.3: SPL correction by linear interpolation between calibration points

To facilitate the interchanging and distribution of calibrations we have designed a simple, human-readable and -editable file format. This XML-based format is described in section D.3. As shown in figure 6.20, `Calibration` instances are created by the `CalibrationFactory`. This is done by parsing a calibrations file which can come from three sources. One is included in the apps themselves and `CalibrationFactory` uses a `ResourceReader` to load it (see section 6.5.3). Every time a new version of either NoiseTube Mobile app is released we include the latest file. However, to enable us to add support for new models without having to release new app versions, and to reach users who keep on using an outdated version, the latest calibrations file is also hosted on the NoiseTube website [376]. If the app can access the Internet, `CalibrationFactory` uses the `download-Calibrations()` method of `NTWebAPI` (see figure 6.23 below) to download it. That file is then used instead of the compiled-in one and is also saved locally, such that later on, if the Internet cannot be accessed, the previously downloaded file

```
private double correctSPL(double spl)
{  //INPUT = phone; OUTPUT = reference; points sorted in increasing order
   int i = 0;
   while(i < calibrPoints.length && spl > calibrPoints[i][INPUT_IDX])
      i++;
   if(i == calibrPoints.length)
      i--; //use last two calibration points
   double x0, y0, x1, y1; //INPUT -> x; OUTPUT -> y
   if(i == 0)
   {  //interpolate between the origin and the first (0th) calibration point
      x0 = 0.0d;
      y0 = 0.0d;
   }
   else
   {  //interpolate between the (i-1)th and (i)th calibration point
      x0 = calibrPoints[i-1][INPUT_IDX];
      y0 = calibrPoints[i-1][OUTPUT_IDX];
   }
   x1 = calibrPoints[i][INPUT_IDX];
   y1 = calibrPoints[i][OUTPUT_IDX];
   //x = spl; y = return value (i.e. the corrected spl)
   return ((spl - x0) * ((y1 - y0) / (x1 - x0))) + y0;
}
```

Box 6.4: The `correctSPL()` method of the `Calibration` class

can be used. Hence, the compiled-in calibrations file is only actually used when no Internet access is possible and there is no previously downloaded file.

The `getCalibration()` method of `CalibrationFactory` handles the task of picking a suitable `Calibration` to use for the phone on which the app runs, based on the brand and model. If there is no match for the exact model, a calibration for another model of the same brand is returned, and when there are no calibrations from the brand an overall default calibration is used[70]. For this purpose the `Calibration` class and the calibrations file format foresee fields to mark a calibration as the `brandDefault` or `overallDefault`. Additionally there is a field that indicates the "credibility" of the calibration[71], which refers to the circumstances in which the calibration points were determined, whether or not the values have been independently verified, whether or not the calibration matches the model, only the brand or neither, etc. As mentioned in section 6.3.1 the credibility of the used calibration is stored/transmitted along with SPL measurements.

In section 6.6.2 we discuss the correction methods applied in SLM apps developed by others. In chapter 7 we return to the matter of determining calibration points for NoiseTube Mobile and the accuracy it can achieve on calibrated phones.

This concludes our discussion of the classes involved in sound level measuring.

[70] It is reasonable to assume that the microphones of different models of the same brand may have similar characteristics, and that all mobile phone microphones are similar to some extent. Hence it is generally better to use the calibration for another model/brand than to use no calibration at all.

[71] The possible values of the credibility field are documented in section D.3.

6.5.4.3 Geo-tagging

The diagram in figure 6.21 shows the classes that implement the geo-tagging functionality (see section 6.3.2). We used the Abstract Factory pattern because the platforms' positioning APIs and coordinates classes are incompatible. Again, `NTClient` and its subclasses play the role of respectively the «AbstractFactory» and the «ConcreteFactories». The `GeoTagger` class is an «AbstractProduct», subclassed by the «ConcreteProducts» `JavaMEGeoTagger` and `AndroidGeoTagger`, which rely on the APIs offered by the respective platform to start and stop listening for new GPS coordinates[72].

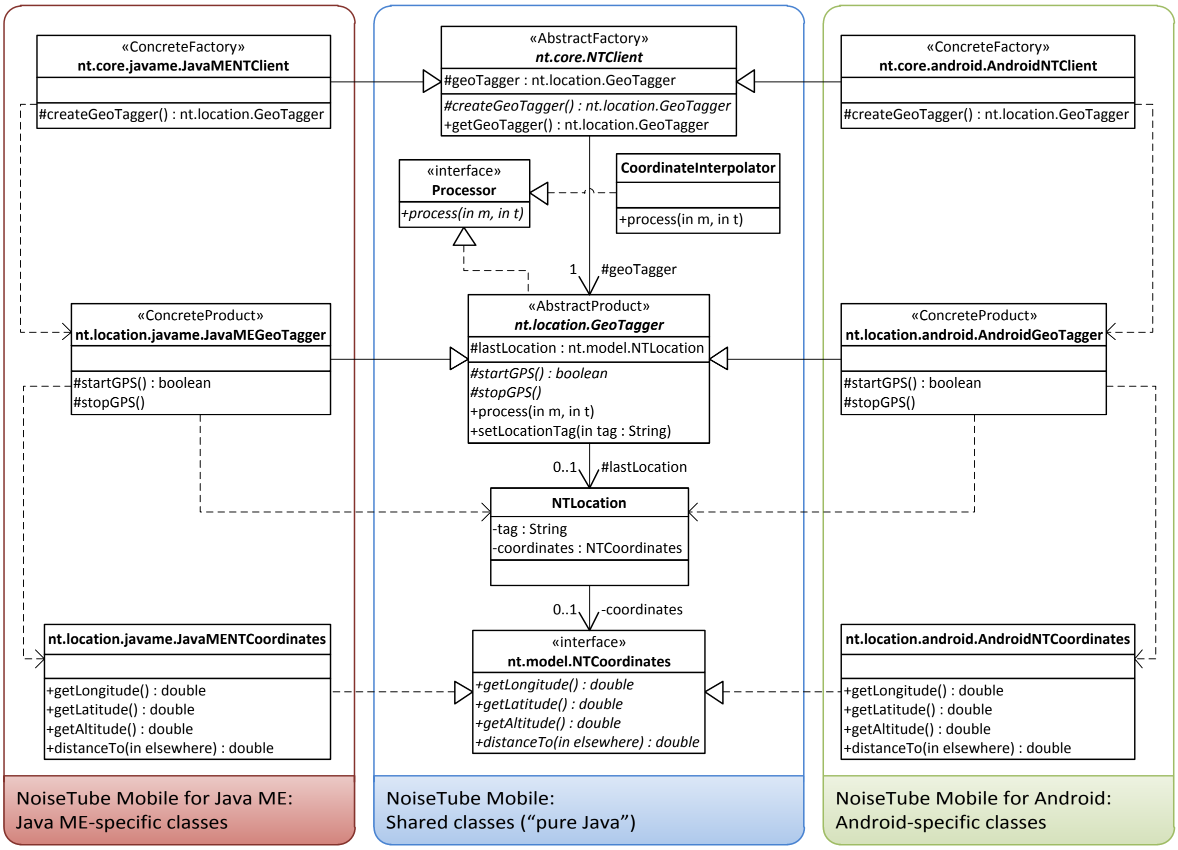


Figure 6.21: Geo-tagging classes

Every time new GPS coordinates are obtained they are stored in a new instance of either `JavaMENTCoordinates` or `AndroidNTCoordinates`, which is in turn stored in a new `NTLocation`. The latter then becomes the new value of the `lastLocation` field of `GeoTagger`. As noted in section 6.3.2, the Java ME app allows users to geo-tag measurements with manually typed a location tag. When this happens the UI calls `GeoTagger`'s `setLocationTag()` method. Here the tag is either added to the `lastLocation` if there is one and it is not too old, or it is stored in a new `NTLocation`

[72] In their dealings with underlying positioning APIs these classes specify sharp timing and accuracy criteria to force the use of GPS (rather than an alternative technology), and to be able to (ideally) tag each new measurement with new coordinates, although that is not always possible (see section 6.5.5.2).

instance (without coordinates) which is then set as `lastLocation`. The `GeoTagger` class implements the `Processor` interface. Every time its `process()` method is called, the new measurement is tagged with the `lastLocation`, unless it is too old. Up to 20 consecutive measurements can be tagged with the same coordinates until they are discarded (location tags can be reused indefinitely). The purpose of the `Coordinate-Interpolator` class shown in figure 6.21 is discussed below in section 6.5.5.2.

6.5.4.4 Social & automatic tagging

As discussed in section 6.3.3 the Java ME app only allows users to tag one measurement at the time, while the Android app lets them to tag measurement intervals instead. The former functionality is provided by the `SingleMeasurementTagger` class, which implements the `Processor` interface as shown in figure 6.17. This class receives a `String` with tag(s) and an associated timestamp from the UI via its `setTags()` method and passes them on to the most fitting measurement (closest to the timestamp) the next time its `process()` method is called. The `Measurement` instance receives the tags in its `addUserTags()` method, where the comma-separated string is split such that each tag can be stored separately. When the user tags measurements in the Android app the UI creates a new `TaggedInterval` instance, in which the tag(s) are stored, and passes it on the currently running `Track` using the `addTaggedInterval()` method.

Although most of the "learning" stage actually happens on the server-side (i.e. through automatic tagging by the CM, see section 5.5.1.1), we have also foreseen a way for data to be automatically analysed and subsequently tagged on the client-side. As discussed in section 6.3.4, there are currently two such features, which are respectively implemented by the `HighExposureTagger` and the `PeakTagger` classes[73]. Both implement the `Processor` interface and analyse and possibly tag measurements in their `process()` method. To keep automatic tags separate from user (i.e. social) tags they are passed to the `Measurement` instance using the `addAutomaticTag()` method. If we would add additional automatic taggers in the future these could also create `TaggedIntervals`.

6.5.4.5 User interface

The implementation of the UI cannot happen in the shared codebase because, as explained in section 6.5.1.1, the UI toolkits offered by Java ME and Android are completely different. Therefore, the measuring pipeline classes (most notably `Track` itself), which are part of the shared codebase, can only interact with UI through the `TrackUI` interface, shown in the diagram in figure 6.22.

[73] Even though these classes are part of the shared codebase they are only activated in the Java ME app.

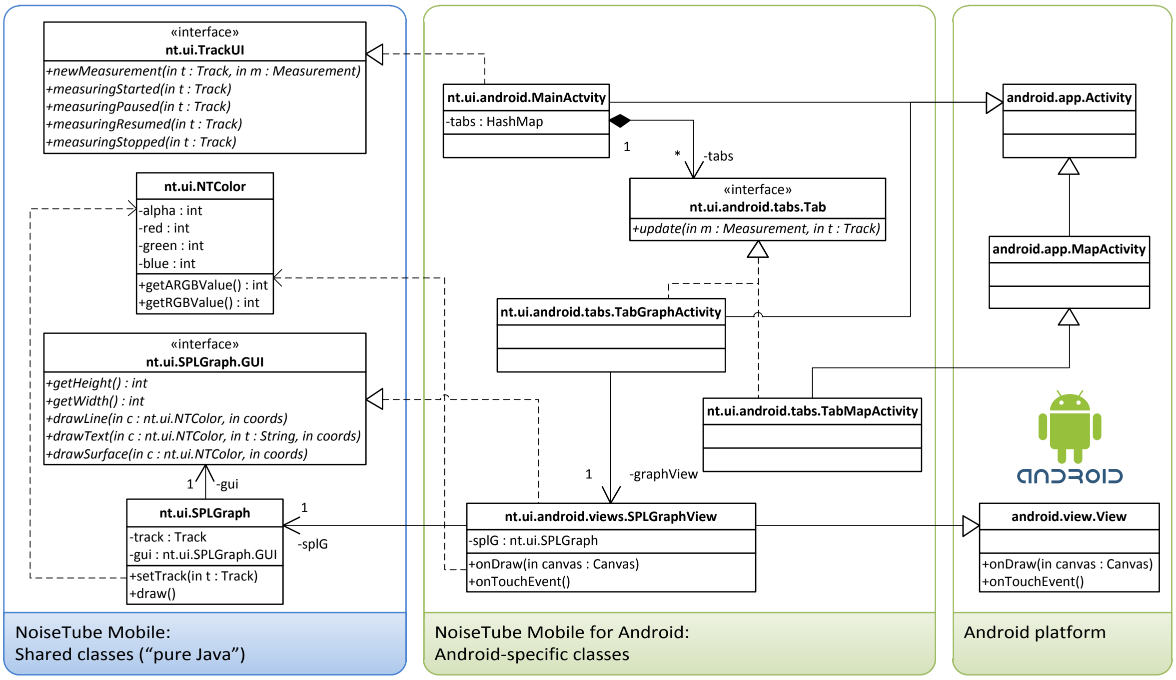


Figure 6.22: User interface classes, showing only shared and Android-specific parts

We will not go into the details of the implementation of the UI of both apps, except for one aspect: the drawing of the SPL graph shown in figure 6.2. Despite the difference between the platforms, we have implemented this on an abstract level, such that the code could be shared and to ensure that the graph looks same in both apps. Figure 6.22 illustrates how this is done on Android, the situation in the Java ME app is analogous.

The shared drawing routine is contained in the `draw()` method of the `SPLGraph` class. This method constructs the entire chart: both axes, the decibel labels and coloured lines, the plot of the measurements currently held in the buffer of the `Track` and the indications of tagged measurements and intervals. While the `draw()` method handles all pixel-level calculations the actual drawing requires platform-specific APIs and is therefore delegated to an instance of the `SPLGraph.GUI` interface, which declares primitive operations for the drawing of lines, text and surfaces. Because both platforms use a different way to express colour values we have introduced the `NTColor` class to pass colours to the drawing operations. The `SPLGraph.GUI` interface is to be implemented by a platform-specific widget class. In the case of the Android app this is `SPLGraphView`, which is a subclass of `View`, the super class for all Android widgets. Upon being instantiated the platform-specific widget creates an instance of `SPLGraph` and passes a reference to itself, typed as `SPLGraph.GUI`, such that both instances hold references to one another. Whenever the widget gets the order to update itself (on Android this means the `onDraw()` method is called), it calls `SPLGraph`'s `draw()` method do redraw the graph. These interactions are similar, yet not identical, to the *Adaptor* [198: pp. 139–150] and *Template Method* [198: pp. 325–330] design patterns.

6.5.4.6 Data saving

As discussed in section 6.3.5, NoiseTube Mobile supports three data saving models: storage in local files (one per track), submission to the NoiseTube CM, and disabled saving. The diagram in figure 6.23 shows the classes that are involved in this functionality.

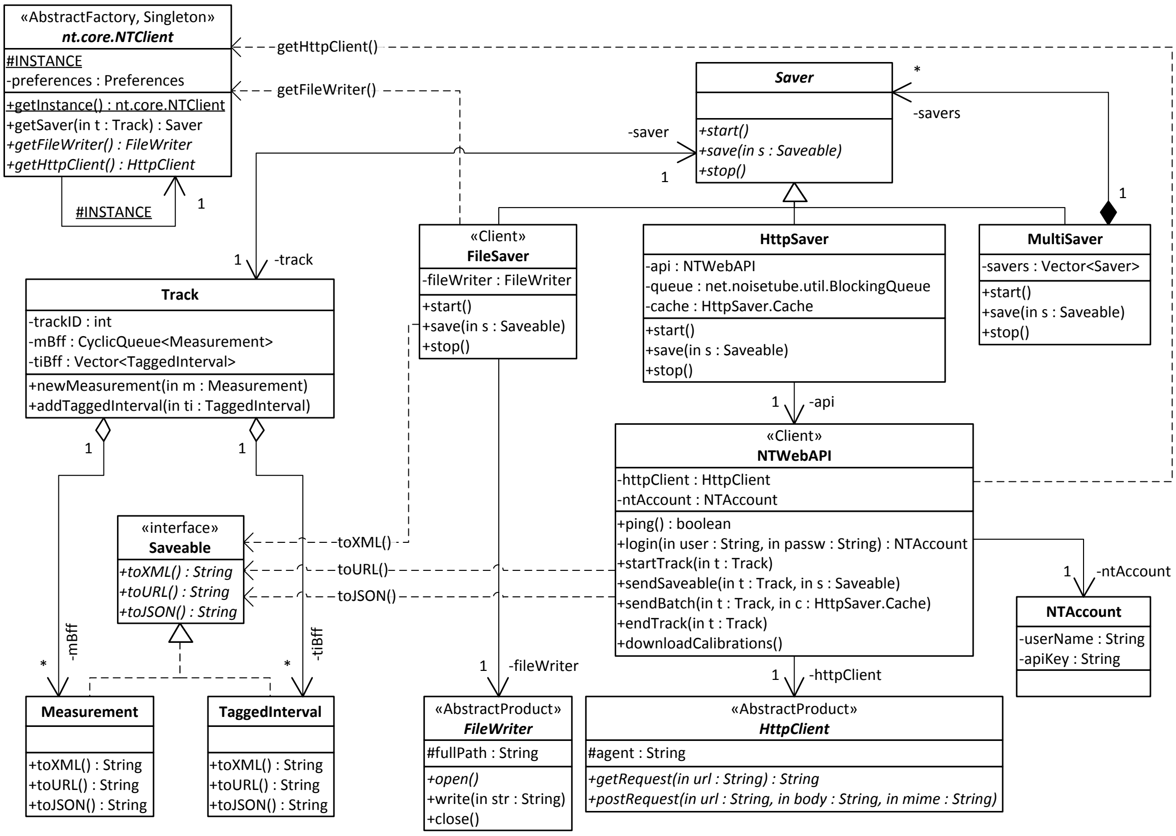


Figure 6.23: Data saving classes

When a new `Track` instance is created it will request a `Saver` instance from the `NT-Client` via the `getSaver()` method. What is returned by the `NTClient` depends on the user's data saving preference (as stored in a `Preferences` instance)[74]:

- If the user has selected local storage an instance of `FileSaver` is returned;
- If the user has select to submit data to the CM a `MultiSaver` instance, containing a `FileSaver` and an `HttpSaver` instance, is returned;
- If the user has select to not save data a `null` pointer is returned.

As explained in section 6.5.4.1, every time a `Track` needs to make room in its buffer the oldest `Measurement` is removed and saved. This means it is passed to the `Saver` via the `save()` method. Moreover, every time it receives a new `TaggedInterval` the

[74] As noted in section 6.3.7.1, at some point we also supported submitting data via SMS, which was implemented by the now deprecated `SMSSaver` class.

Track immediately passes it on to the Saver via the same method. Both Measurement and TaggedInterval implement the Saveable interface, which obliges them to implement the toXML(), toURL() and toJSON() methods. Each of these returns a serialised String representation of the model instance, respectively formatted as an XML node, a series of URL parameters, or a JSON expression.

The FileSaver class saves Saveables to a locally stored track file using the XML format described in section D.2. The class obtains XML node representations of Saveables by calling their toXML() method. Because creating and writing data to files requires platform-specific APIs the FileSaver delegates that to a FileWriter instance, which it gets from the NTClient factory class, as discussed in section 6.5.3.

To implement the backup feature mentioned in section 6.3.5, which requires that while submitting data to the CM is also saved to a local file, we have used the *Composite* design pattern [198: pp. 163–173]. The role of the «Composite» is played by the MultiSaver class, which holds a collection of other Saver instances – i.e. its «Components» – and passes on all commands – e.g. start(), save(), stop(), etc. – to each of them.

The HttpSaver class submits Saveables through the Internet. However, to make it easier to let future NoiseTube Mobile derivatives submit data to other Web services, we implemented the actual interaction with the NoiseTube CM in an separate class, namely NTWebAPI. In turn, NLWebAPI delegates the actual HTTP communication to an HttpClient instance because this requires platform-specific APIs. As discussed in section 6.5.3, the HttpClient instance is obtained from the NLClient factory class. To avoid slowing down the whole application data submission happens asynchronously: in its save() method the HttpSaver adds each Saveable to a queue, from which they are taken out by a separate thread that handles the actual submission[75]. As explained in section 6.3.5, we have also taken precautions to avoid that unstable network connections cause data to be lost. Concretely, HttpSaver has two operating modes: real-time and batch submission. In the former, which is the default, Saveables are sent one at the time via NTWebAPI's sendSaveable() method. In latter, Saveables are first cached and later sent in batches of (multiples of) 30 via NTWebAPI's sendBatch() method. If real-time submission fails, we switch to batch mode to buy time for the connection to be restored. If submission of a batch succeeds we switch back to real-time mode, if it fails we stay in batch mode and cache the next 30 Saveables before trying again.

The NLWebAPI class uses the API described in section D.1 to authenticate the user, start and end tracks, and send Saveables. Upon successful authentication the login() method returns an NLAccount instance containing the user's API key which is needed to submit data. The NTAccount in then persistently stored via Preferences such that the next time the used does not need to enter his/her username and password. In

[75] FileSaver works synchronously because local file I/O is much faster than HTTP communication.

the `sendSaveable()` method, a `Saveable` is encoded – using parameters obtained by calling `toURL()` – in the URL of an HTTP `GET` request, executed using `HttpClient`'s `getRequest()` method. In the `sendBatch()` method, the contents of the cache are encoded as a single JSON expression (assembled by calling `toJSON()` on each `Saveable`) which is then sent in the body of an HTTP `POST` request, executed using `HttpClient`'s `postJSONRequest()` method.

6.5.5 Dealing with bugs & limitations

During the development and maintenance of NoiseTube Mobile we have come across some problematic device-level bugs or limitations. Because these issues cannot be fixed at the application level we have had to devise ways to work around them. One example, the sampling rate bug on some Sony Ericssons, was already covered in section 6.5.4.2. Here we discuss two others.

6.5.5.1 Memory leak in Nokia's MMAPI implementation

The Java ME variant of NoiseTube Mobile is especially used on Nokia devices running Symbian [519]. Unfortunately the Java ME runtime environment on these phones [380] contains a bug that has caused us considerable trouble. The problem is a memory leak in the MMAPI [283] implementation which causes the runtime to run out of memory after an app has been recording audio for some time. When this happens, the app either freezes, crashes or displays an "out of memory" error and then exits.

Through extensive testing we found out that on a given model, the problem always occurs after more or less the same time, and that this running time depends on the sampling rate (i.e. when recording at a lower rate it takes longer for the problem to occur). Moreover, if the app is manually exited and restarted (before the problem occurs) the "timer is reset" (i.e. the running time is not shortened by the previous run of the app). All of this confirms that the problem is indeed caused by a memory leak. Later we found out that others had run into the same problem when recording audio on Nokia phones [292, 351], confirming that it was not caused by our own code. As far as we have been able to check, even the latest Symbian devices from Nokia still have this problem.

As the bug is situated in the runtime environment itself, it cannot be easily avoided, fixed or bypassed at the Java level. To work around it, others have either partially [351] or completely [292] implemented their apps in C++ instead of Java. Although we have also briefly experimented with a partial implementation in C++[76], we have not further pursued

[76] Similarly to [351], our approach was to do the audio recording in a native Symbian application (written in C++) which communicates with the MIDlet (written in Java) through a local socket connection.

this because it makes the distribution, installation and usage of the app more complex (because it consists of two parts), and would be incompatible with non-Symbian Java ME phones. Although a full rewrite in Symbian C++ might have been the best solution we did not pursue it because we rather spent time on porting NoiseTube Mobile to more modern platforms such as Android and iOS.

Because we had already invested in a series of Nokia devices to conduct experiments with, we were obliged to find a workaround for the problem. As a first step we have experimentally determined "safe" running times for various sampling rates[77]. We did this on a fairly cheap, low-end device (the Nokia 5230 [386]) to avoid overestimating running times for other devices[78]. For example, at 16 kHz the running time is ± 19 minutes, but at 48 kHz, the highest supported rate, it is only about 7 minutes. With this information we have implemented two slightly different workarounds. Both are based on a timer that counts down the running time for the sampling rate that is being used, such that appropriate action can be taken *before* the app freezes or crashes.

For the experiments discussed in chapter 7, in which we wanted to evaluate and improve data quality, lowering the sampling rate to stretch the running time was not an option, so we stuck with 48 kHz. To allow tracks longer than 7 minutes to be made, we have implemented a mechanism that, when the time is up, saves the program state to a temporary file, schedules the app to restart automatically a few seconds later, and then exits properly. When the app restarts, the state file is read and measuring continues as if nothing happened. While this does not exactly result in a seamless user experience (i.e. the app disappears from the screen for up to 10 s), it requires no user intervention and we informed the experiment participants such that they knew what to expect. The API for scheduling automatic (re)starts of a MIDlet requires user permission[79]. However, this permission is, at least on Nokia devices, mutually exclusive with that to access the Internet (i.e. users cannot grant both at the same time). Consequently this workaround rules out communication with the NoiseTube CM and thus forces data to be saved to track files. While unfortunate, this was not problematic for our experiments, because most analysis could only happen after the measuring campaign had finished.

This solution is unsuitable for the app that is offered for download, because Nokia owners would no longer be able to submit data to the CM from within the app, and automatic restarts would likely be confusing. Therefore, since v2.0.0 (released in April 2011) the publicly available Java ME app includes a different workaround. We have taken a pragmatic decision to use a lower sampling rate to stretch the running time. Concretely, we use 16 kHz instead of the highest rate supported by the device. This means audio signal fidelity is traded in for running time, because, as formulised by the Nyquist–Shannon

[77] By which we mean a conservative estimate of the duration the app can be expected function properly (i.e. not crash or freeze) on an affected device while recording audio at a specific sampling rate.

[78] On pricier devices with more RAM it may take longer before available memory runs out.

[79] Refer to section E.3.4 for an explanation of MIDlet permissions.

sampling theorem (see section A.4.2.1), when sampling at 16 kHz one can only capture frequencies up to 8 kHz (instead of 24 kHz at 48 kHz). When the time is up, the app automatically stops measuring and exists, but not before informing the user of the reason and suggesting him/her to manually restart it. If the app is restarted within 5 minutes it will continue with same track instead of starting a new one. We should note that the lowering of the sampling rate and the timer function are only part of the Java ME app (not the Android one) and are only activated when it actually runs on a Nokia device.

6.5.5.2 Slow GPS updates

As mentioned in section 6.5.4.3, the platform-specific subclasses of `GeoTagger` ask the underlying positioning API to supply GPS coordinates at a rate that allows to tag each new SPL measurement with new coordinates. This requires an update rate of about 0.5 Hz in the Java ME and 1 Hz in the Android app. However, GPS receivers in mobile phones sometimes have trouble keeping up with this rate, especially if signal reception is poor (e.g. in urban canyons [252]). When this happens there are two possible scenarios. One is that `GeoTagger` will reuse previously received coordinates (up to 20 times) because it has not yet received new ones. The other is that the device itself reuses coordinates to be able to send location "updates" to the app at the requested rate[80]. Either way, we end up with a track that contains series of measurements with identical coordinates. When such a track is visualised on a map, identically geo-tagged measurements are drawn on top of each other. When the track was made while walking, the map could give the impression that, instead of keeping a constant pace, the user stood still for a while, then took a big leap to a place some 30 m away, then stood still again, etc.

So solve this problem, and hence produce more realistic maps, we have implemented the `CoordinatedInterpolator`, shown in figure 6.21. This `Processor` class scans the buffer of the running track for series of measurements with repeated (or missing) coordinates, and replaces these with interpolated ones. For example, representing coordinates by integers, the series [7, 7, 7, 4] and [7, `null`, `null`, 4] would both become [7, **6**, **5**, 4]. To interpolate coordinates we divide the distance between the series' end points (which remain unchanged) according to the exact timestamps[81] of the intermediary (to be interpolated) measurements. We implemented two algorithms for this. One uses Euclidean distance to interpolate coordinates along a straight line. The other uses the great-circle distance and course [585] to interpolate coordinates along an arc over the surface of the WGS84 ellipsoid model of the earth[82]. The current app versions use the second algorithm because it is more accurate, although also more computationally intensive.

---

80 This technique is called *point clamping*, and may be implemented at the level of the GPS chip or the OS/middleware. The Sony Ericsson W995 [486] is one device which applies point clamping.

81 I.e. we assume a constant travel speed, but not necessarily a constant distance between points.

82 WGS84 is the reference system used by GPS coordinates [597].

6.5.6 Possibilities for reuse and extensions

In section 6.5.1.4 we noted that facilitation of reuse and extensions was a concern during the design and refactoring of NoiseTube Mobile. Here we briefly look at some options:

SLM library

We have ensured that the coupling between the classes involve in sound level measuring (see section 6.5.4.2) and the rest of the apps' source code is very low. Hence, it is straightforward to reuse these classes in other contexts. To encourage this we have packed them as the *NoiseTube SLM Library for Android*. This library, which will soon be published on *Google Code* [378] under the same LGPL license, allows SLM functionality to be added to any Android app with just a few lines of code[83]. It works independently from the rest of the NoiseTube system except for the downloading of calibration settings (which can be disabled). At the Software Languages Lab we have recently used this library to add SLM functionality to the Android client of the *SenseTale* system, a platform for DIY-style development of sensor-driven applications created by Alcatel-Lucent Belgium [10] in the scope of the DiYSE project [132], in which our lab was also involved.

Additional `Processors`

The `Processor` interface makes it simple to add classes that analyse, annotate, correct or otherwise process individual or series of measurements. For instance, a new `Processor` could monitor various sensors and use machine learning algorithms as described in [350–352] to infer the phone context and user activities and save that information as automatic tags. By simply extending the `Measurement` class with a field for an `AudioStream` and adding one line in the `SoundLevelMeter` class it would because possible for `Processors` to apply further analysis to the audio signal. A promising usage could be (semi)-automatic classification of sound sources, as demonstrated on smartphones by Lu et al. [327].

Additional sensor inputs

The measuring pipeline (see section 6.5.4), is also flexible regarding the addition (or replacement) of sensor inputs. Currently the driving input is the `SoundLevel-Meter` class but the `Track` could just as well receive `Measurements` from one or more other sources. For instance, students working under our supervision have extended NoiseTube Mobile with support for external weather or air quality sensors (see chapter 8). To handle such new types of data the `Measurement` class can be extended, subclassed, or, even better, an abstract superclass can be extracted such that different types of measurement become siblings. None of the `Saver` classes have to be changed to deal with the new data because `Saveables` are responsible for generating `String` representations of themselves (see section 6.5.4.6).

[83] It would be trivial to make an equivalent library for Java ME if there would be an interest for that.

6.6 Related work

Here we discuss the main examples of related work and compare it with our own research and its artefacts. First we look at projects and/or mobile apps that are to some extent similar to NoiseTube (Mobile) in purpose – i.e. the use of mobile phones as sound level meters and tools to assess environmental noise in a participatory fashion. Next, we look at how these apps tackle the problem of calibration.

6.6.1 Phones as SLMs and noise mapping tools

We were not the first, nor the last, to have the idea to use mobile phones to measure environmental noise. In their influential paper from 2006, Burke et al. already suggested that participatory sensing could be applied to let « *citizens* [...] *join a data-collection campaign to document noise levels in a community* » [71: p. 4]. Yet when the NoiseTube project started in the spring of 2008 we were not aware of any working solutions. However, as it turned out, ours was not the only effort to change that. For instance, early on we learned that a group at Cambridge University [542] was doing similar work (see below). Since then, several researchers have launched similar projects. Moreover, the boom of the market for smartphones and apps has inspired people outside academia to also develop (mostly) simple SLM apps. Below we give a chronological overview of what we consider to be the most relevant related work, especially within academia. Table 6.3 summarises the main characteristics of each project or application in comparison with ours.

MobSens / NoiseSPY

MobSens, first called *MobGeoSen*, was an experimental mobile sensing system developed by the universities of Nottingham and Cambridge [293, 294]. It used external sensors to assess air quality and allergen levels and measures the ambient sound level through the phone's built-in microphone. This was most likely the first system to measure SPL on a mobile phone. Later that functionality was put in a dedicated mobile app for the Symbian platform, called *NoiseSPY* [292, 293]. NoiseSPY records audio at a sampling rate of just 8 kHz [292], even though the devices the creators have used to test it support rates up to 48 kHz. This choice is not motivated, which is remarkable because it may severely lower the accuracy of the SPL measurements[84]. No details are provided about how the system was calibrated or how corrections are applied. In [292] the authors state that the deviation between NoiseSPY and a reference SLM is « *very small* », but this claim is not backed with an error estimate. The papers show screenshots of the maps

[84] The reason is that the frequencies that can be captured are effectively limited to 4 kHz or less, covering only about 20% of the human hearing range (see section A.4.2.1).

the MobSens system can generate, but little information is given about the central (server) infrastructure. NoiseSPY is not available for download[85] and hence there is also no publicly accessible platform to share, aggregate or visualise data.

BikeNet

BikeNet, by Eisenman et al. of Columbia University and Dartmouth College, was an experimental mobile sensing system aimed for cyclists [147, 149, 150]. Using a GPS-equipped smartphone and various external sensors attached the bicycle or the cyclist, BikeNet monitored parameters such as covered distance, speed, calories burned, roughness of the terrain, CO_2 levels and the noisiness of the surroundings. Data could be submitted to a simple Web portal called *BikeView* to visualise routes and sensor measurements on a map [147]. Very little information is given about how SPL measuring was implemented and it is unclear whether efforts were undertaken to evaluate or improve (e.g. through calibration) measurement accuracy.

Santini et al.

As an extension of their work on monitoring of environmental noise by means of dedicated wireless sensor networks [180, 460] – mentioned in section 4.3.2.8 – Santini et al. from ETH Zurich also investigated the use of mobile phones in this context [458, 459]. They focused solely on an experimental evaluation of the achievable SPL measurement accuracy. Hence usability aspects were largely left aside – for instance, concerns such as privacy, transparency and context-detection are mentioned but not addressed. Moreover, they did not develop an infrastructure for sharing, aggregating and visualisation of data. The authors developed two SLM apps, one written in Java (ME) and the other in Python, neither app has been made available to the public. Both apply A-weighting and Fast-time weighting (see section A.5.1.3.1). The Java ME app recorded audio at 41 kHz[86] while the Python app could only record at 8 kHz due to API limitations. The apps were tested on three Nokia N95 8GB [385] phones running side-by-side with a professional Class 2 SLM in a laboratory environment using different test signals (white noise, pure tones and a recording of a traffic jam). The authors report on many of the same problems we discussed in section 6.5.1.3, such as the influence of the sampling rate, the presence in the audio recording path of signal processors which may or may not be bypassable, and the differences between the audio recording APIs on different platforms. They note that to achieve more accurate results the sampling rate must be high enough (the Java ME app performed much better than the Python one) and processing steps such as *low-pass* filters should be bypassed if possible.

[85] In 2008 a rudimentary test version was briefly offered for download, but was later taken down.

[86] As this is not a common sampling rate for audio signals we think this was probably 44.1 kHz.

Project / App name	Creators	Type of initiative	Project started in	Project on-going	Open source
MobSens / NoiseSPY	Kanjo, University of Cambridge	Research	2006	Yes(?)	No
BikeNet	Eisenman et al., Dartmouth College	Research	2007	No	No
NoiseTube / NoiseTube Mobile	**Sony CSL Paris & VUB BrusSense**	**Research**	**2008**	**Yes**	**Yes**
n/a (Santini et al.)	Santini et al., ETH Zurich	Research	2008	No	No
WideNoise	WideTag / EveryAware project (since 2011)	Commercial / Research	2008/2011 (v3.0)	Yes	Yes (since v3.0)
Ear-Phone	Rana et al., University of New South Wales	Research	2009	No	No
CycleSense / Biketastic	Reddy et al., CENC, UCLA	Research	2009	Unclear	No
OpenNoiseMap / NoiseDroid	Foerster et al., University of Münster	Research	2010	Yes	No
da_sense / NoiseMap	Schweizer et al., TU Darmstadt	Research	2011	Yes	No
Eye on Earth / NoiseWatch	European Environment Agency & Microsoft	Authority	2011	Yes	No
AirCasting	HabitatMap & Lunar Logic Polska	NGO	2011	Yes	Yes

(a) Project details

Project / App name	Mobile app platform	App available for download	Annotations	A-weighting
MobSens / NoiseSPY	Symbian (C++)	Not anymore	No(?)	Yes
BikeNet	Symbian	No	No	No
NoiseTube / NoiseTube Mobile	**Java ME, Android, iOS**	**Yes**	**Yes (social tagging)**	**Yes**
n/a (Santini et al.)	Java ME / Symbian (Python)	No	No	Yes
WideNoise	iOS, Android (since v3.0)	Yes	Yes (since v3.0)	No
Ear-Phone	Java ME	No	No	Yes
CycleSense / Biketastic	iOS, Android	Yes	Yes (notes, photos, videos)	No
OpenNoiseMap / NoiseDroid	Android	Yes	Yes (rating, tags, notes)	No
da_sense / NoiseMap	Android	Yes	Yes (categories, label)	No
Eye on Earth / NoiseWatch	iOS, Android, Windows Phone	Yes	No	No
AirCasting	Android	Yes	Yes (notes, tags & photos)	No

(b) Technical details (1)

Project / App name	Calibration/correction features	Calibration/Correction	Data sharing features	Noise mapping features	Publicly accessible community platform
MobSens / NoiseSPY	Mentioned but no details given	Unclear	Unclear	Yes	No
BikeNet	No details given / no correction applied	No	Yes	Yes	No (only as a demo)
NoiseTube / NoiseTube Mobile	**Linear interpolation, built-in & downloadable settings**	**Yes**	**Yes**	**Yes**	**Yes**
n/a (Santini et al.)	Possibility of trimming was evaluated	n/a	No	No	No
WideNoise	v3.0: linear interpol., same setting used for all phones	No	Yes	Yes	Yes (everyAware)
Ear-Phone	Trimming, no built-in settings	Yes	Unclear	Yes	No
CycleSense / Biketastic	No details given / no correction applied	No	Yes	No(?)	Yes
OpenNoiseMap / NoiseDroid	No details given / no correction applied	No	Yes	Yes	Yes
da_sense / NoiseMap	User-accessible trimming setting (?), no built-in settings	Yes	Yes	Yes	Yes
Eye on Earth / NoiseWatch	No details given / no correction applied	No	Yes	Yes	Yes
AirCasting	User-accessible trimming setting, no built-in settings	Yes	Yes	Yes	Yes

(c) Technical details (2)

Table 6.3: Comparison of NoiseTube (Mobile) and similar projects/apps created by others

WideNoise

WideNoise is an app developed, initially only for the iPhone, by a company called WideTag [573]. Rather than making one SPL measurement after the other, like NoiseTube Mobile does, WideNoise makes one at the time (i.e. the user has to click a button every time). Measurements are computed over 5 s intervals. To help users interpret the decibel values the app associates each measurement with a prototypical example (e.g. a sleeping cat for 40–50 dB, a car for 70–80 dB, etc.)[87].

[87] Similarly to the examples we list in table A.1 on page 271.

Measurements are geo-tagged and can be posted to the WideNoise website, where they are shown on a map, or to social networks. In mid-2011, development of a new version was started under impulse of the *EveryAware* project [175]. In this v3.0, which is also available for Android, annotation features were added. Users can rate measurements along 4 dimensions: love vs. hate, calm vs. hectic, alone vs. social, and nature vs. man-made. All data is now also sent to the EveryAware community platform. The source code of WideNoise v3.0 for both iOS and Android has been released under an open source license. By reviewing the code we found out that WideNoise applies a linear interpolation algorithm to account for microphone sensitivity, similarly to – and quite possibly inspired by – NoiseTube Mobile. However, the apps for both platforms (iOS and Android) "correct" measurements using the same, hard-coded array of calibration points, no matter on which device they run. Hence, although we have not done any side-by-side comparisons, we have serious doubts about the accuracy these apps can achieve. The WideNoise creators have not provided figures on the (estimated) accuracy.

Ear-Phone

Ear-Phone is an experimental mobile sensing system aimed at noise mapping and was created by Rana et al. from the University of New South Wales, Australia [438]. In [438] the authors provide a fairly detailed technical description of their system, which is in many ways similar to ours. Like NoiseTube Mobile, the Ear-Phone app computes series of $L_{\mathrm{Aeq,1s}}$ values from A-weighted audio samples and constantly updates an overall $L_{\mathrm{Aeq},T}$ value (where T is the duration of the measuring session). Measurements are corrected using a constant offset, to be determined per phone or model. As we will discuss below, this is sometimes called "trimming". The authors have evaluated the accuracy of their system in different circumstances. For (pre-recorded) roadside noise they achieved an accuracy within 2.7 dB of a reference SLM while the phone was held in hand. The most innovative feature in the Ear-Phone architecture is the use of a technique called *compressive sensing* [437] by the central server to reconstruct sound levels at different places/times from aggregated data coming from multiple participants. The Ear-Phone system has only been tested on a small scale and the app has not been offered for download, hence there is also no publicly accessible community platform.

CycleSense / Biketastic

Biketastic [81, 443], developed by Reddy et al. of the CENS group at UCLA in the scope of the *CycleSense* project [83], is another mobile sensing system for cyclists. It aims to help them to document and share biking routes. The Biketastic app uses the phone's GPS to infer distance and speed, and combines the accelerometer and microphone to assess the roughness of the terrain and the noise level along the route. Routes can be further documented with notes, photos or videos. Everything can be shared and visualised online [81]. Although the app measures SPL and the online platform can show the maximum, minimum and average level measured along a route, the Biketastic system is not aimed at the assessment of environmental noise as such. No details are provided about the way SPL measuring works, nor about matters like frequency weighting, calibration or accuracy.

OpenNoiseMap / NoiseDroid

OpenNoiseMap is a project by Foerster et al. of the University of Münster [183–185]. With an Android app called *NoiseDroid* people can measure SPL and annotate the measurements with ratings, tags or notes. The data can be submitted to a website that generates noise maps. The creators provide no information on the way SPL measuring is implemented, nor about frequency weighting, calibration or accuracy.

da_sense / NoiseMap

The *Darmstadt Sensor Netzwerk*, shorted to *da_sense*, is an urban sensing platform created by Schweizer et al. of the Technische Universität Darmstadt. The platform, which currently focusses exclusively on the city of Darmstadt, uses WSNs to monitor gas concentrations at different places in the city and a participatory sensing app, called *NoiseMap*, to collect sound level measurements [466, 467]. An interesting aspect of the NoiseMap app, which is only available for Android, is a game-like "achievements" feature that awards users with titles (e.g. "explorer", "expert", etc.) and star ratings depending on the amount of data they collect, whether or not to go into previously uncovered areas, etc. Moreover, the system regularly sends out "challenges" to the users. In [467] the creators mention, but do not motivate, that the app records audio at a sampling rate of 22050 Hz, which is well below the maximum rate supported by most, if not all, Android devices. Moreover, they explain the app computes the L_{eq} of intervals of 0.5 s each and does currently not apply any frequency weighting, although they plan to implement A-weighting in the future. The paper mentions a user-adjustable trimming setting and stresses the importance of easy calibration to increase participation. However, the version of the app that is currently offered for download does not seem to have this setting. No accuracy estimates are provided.

Eye on Earth / NoiseWatch

Eye on Earth is a Web-based platform created by the European Environment Agency, Microsoft and other partners. As already mentioned in section 2.4.3.5, the goal is to enable citizens to explore and reuse environmental data made available by authorities [169]. Under the name *Eye on Earth Watches* the platform has recently been extended with mobile sensing features to allow citizens to not only consume but also contribute data. Three environmental monitoring apps are currently available: *AirWatch*, *WaterWatch* and *NoiseWatch* [170]. The latter app allows users to measure SPL over an interval of 10 s. Like in WideNoise, users have to manually initiate each measurement. After making a measurement it can be submitted, along with GPS coordinates and a time stamp, to the Eye on Earth platform. No details on the SPL measuring algorithm, calibration or accuracy are provided.

AirCasting

AirCasting is a recent initiative of an NGO based in New York [235]. In collaboration with a Polish subcontractor they have developed a noise mapping app for Android. Users can annotate data with notes, tags and photos and can submit it to a community platform. Both the mobile app and the Web platform are open source. Users can calibrate the app with a trimming setting. No accuracy estimates are given.

Other SLM apps

Since the NoiseTube project got underway in the summer of 2008 the market for smartphones and associated apps has grown explosively (see section E.2). This has led several (mostly) independent software developers to create apps that provide (or promise) basic sound level meter functionality. Whether some of them were inspired by NoiseTube or other research projects is hard to tell, yet not unlikely. In a quick survey of the major app stores[88] we found over 30 such apps. Some are free, in which case they usually display ads, others have to be bought. As is typical on app stores, most of free ones also come in a paid-for version without ads and possibly with extra features. Generally these apps are stand-alone, in the sense that there is no way to upload or share data, nor any noise mapping functionality.

The majority of these apps are little more than novelty gadgets, made with little regard for accuracy. Information provided prior to purchase or download is often limited, incorrect, contradictory[89] or poorly formulated. Moreover, some creators resort to inflated promises[90] and questionable sales techniques[91] to attract attention. Since most apps are neither well documented, nor open source, it is hard to verify if SPL measuring is properly implemented. Of the about 15 apps we have personally tested many left us with the suspicion that the creators had a flawed understanding of acoustics and SLMs. For instance, only 3 apps applied frequency weighting, and while some allow the user to switch between a "fast" and "slow" mode most do not apply actual time-weighting (see section A.5.1.3.1) and instead just change the refresh rate of the display. Moreover, although it is simple to implement on smartphones, most apps do not support data logging.

However, there are a few notable exceptions among these poor offerings. For in-

[88] Google Play [219], Apple's iTunes app store [31] and the Nokia Store [382].

[89] For example, one Android app is called *Professional dB (SPL) Meter*, even though the creator notes that « [it] *is intended for personal (not industrial or professional) use only* ».

[90] For instance, the *SPL Meter*, another Android app, is being called « *a professional-grade sound level meter* », as if any professional acoustician would use a €0.69 mobile app as his/her tool of choice.

[91] Most likely to boost sales, one developer, Darren Gates, offers no less than 15 equally-priced SLM apps with virtually identical functionality on Google Play.

stance, *SoundMeter* [177], an iPhone app by Faber Acoustical, and *AudioTool* [33], an Android app by Armchair Applications, are properly documented and offer additional functionalities found on professional SLMs, such as A- and C-weighting, switching between L_{eq}, fast/slow time-weighted SPL and peak sound level[92], frequency spectrum visualisation and analysis, etc. Then again, at respectively €15.99 and €4.72, these two are also considerably more expensive than the other apps[93].

Most apps we tried out included a user-accessible calibration function. Usually measurements are corrected by trimming, although some apps instead use "scaling" or a combination of trimming and scaling. Below we explain what this means. We found only 4 apps with built-in calibration settings for different phone models (never more than 7), none allowed additional settings to be downloaded.

We were not the first, and certainly not the last, to think up, design and develop a mobile sensing system for sound level measuring and assessment of environmental noise. However, as is clear from the overview in table 6.3 and the discussion of other SLM apps above, ours is the most complete solution to date. For instance, our app was the first to introduce social tagging, remains one of the few to support A-weighting and the only one that can be calibrated remotely via downloadable settings. Moreover, we were the first to develop a platform (i.e. the NoiseTube CM) that facilitates sharing, aggregation and exploration of measurement data and annotations (i.e. tags), as well as noise map production. Only the more recent projects, some of which may well have been inspired by ours, seem to have similar attention for the collective side of things. Perhaps most importantly, we were the first to open our solution to the public by allowing anyone to download the app and use the CM, and by actively stimulating participant recruitment[94]. In doing so we have experienced first-hand that, as pointed-out in section 3.3.1.2, managing a viable and growing mobile sensing system and community, involving participants the world over, poses organisational and technical hurdles that should not be underestimated.

In a sense, the fact that the initiators of the projects listed in table 6.3, whether they are researchers, authorities or NGOs, have all developed their own separate system – none of which is technically superior to ours – constitutes a duplication of effort. Perhaps things would have turned out differently if we had released the NoiseTube source code earlier[95], though we cannot be certain of that. On the other hand, the fact that similar projects and apps keep on appearing confirms that the goals we set in the summer of 2008, and the research and development efforts that have followed, remain highly relevant today.

[92] Not to be confused with maximum sound level (see sections A.5.1.4 and A.5.1.5).

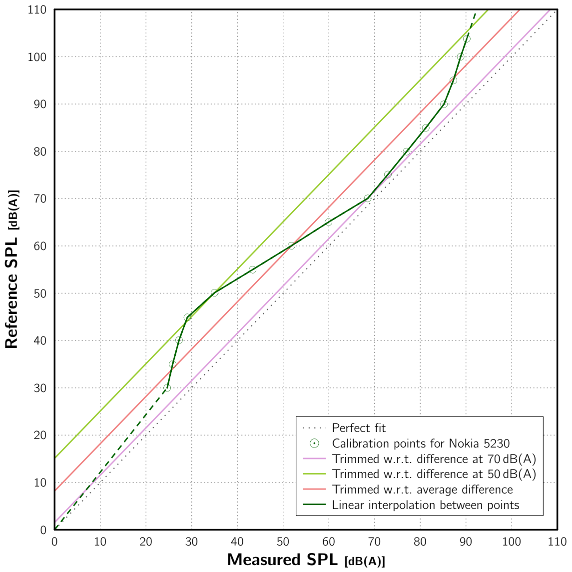
[93] Faber Acoustical also offers a free simplified derivative of SoundMeter called *dB* [176], which lets users share geo-tagged SPL measurements, along with annotations (photo and notes), via e-mail and Twitter.

[94] By presenting at non-scientific events and by reaching out to mainstream media (see appendix G).

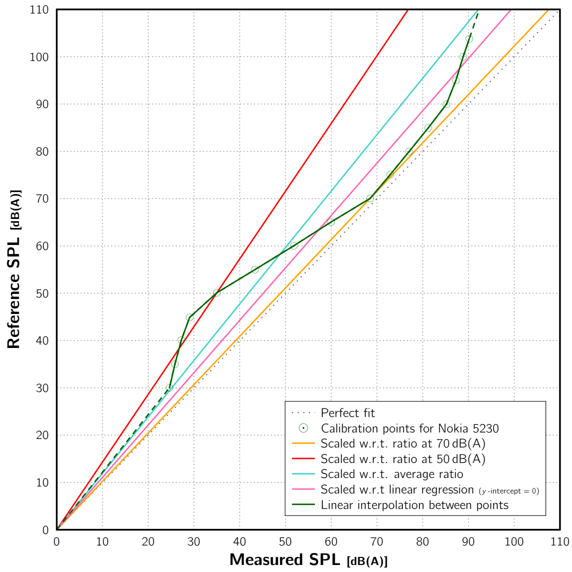
[95] Even though the decision to go open source had been taken at least a year earlier, the code was only released on June 2011 due to pending approval from Sony France.

6.6.2 Calibration & correction methods

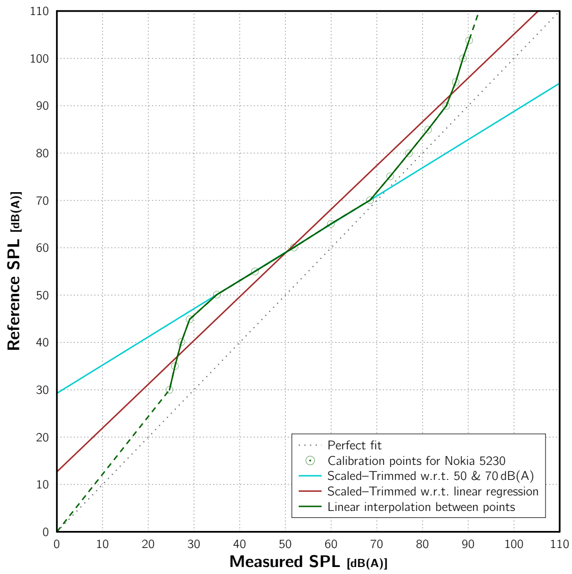
Of the SLM apps discussed in the previous section, only WideNoise corrects measurements in a similar manner as our linear interpolation algorithm described in box 6.3. However, since WideNoise uses a single hard-coded array of calibration points this correction is rather arbitrary and the app cannot be calibrated for specific phone models. Quite a few of the other apps either do not have calibration/correction support at all, or their creators did not mention/document it. In the apps *with* calibration support, measurements are corrected using *trimming*, *scaling* or a combination of both methods.



(a) Trimming



(b) Scaling



(c) Scaling–Trimming

Chart 6.3: Calibration & correction methods found in related work, applied to a Nokia 5230

Trimming

Uncorrected SPL values are adjusted by adding a (possibly negative) offset specific to the device or model:

$$L_p = L_{p,\text{phone}} + \Delta_{phone} \tag{6.4}$$

The Δ_{phone} offset is constant for all levels and frequencies. To choose its value only a single calibration point is needed. For instance, if the mobile app measures 35 dB(A) while the reference indicates 50 dB(A) we get $\Delta_{phone} = 50 - 35 = 15$ dB. Of course it is possible to do comparisons at multiple levels and then take Δ_{phone} as the average difference. Either way, as shown by chart 6.3a the resulting graph is a straight line parallel to the "perfect fit" (i.e. where no correction is needed/applied).

Scaling

Uncorrected SPL values are adjusted by multiplying them with a scale factor specific to the device or model:

$$L_p = S_{phone} \cdot L_{p,\text{phone}} \tag{6.5}$$

The S_{phone} factor is constant for all levels and frequencies. To choose its value only a single calibration point is needed. For instance, if the mobile app measures 35 dB(A) while the reference device indicates 50 dB(A) we get $S_{phone} = \frac{50}{35} \approx 1.43$. Alternatively when comparisons at multiple levels are made S_{phone} can be taken as the average ratio, or determined by means of linear regression (with y-intercept $= 0$). Either way, as shown by chart 6.3b the resulting graph is a straight line through the origin and with a slope of S_{phone}.

Scaling–Trimming

Uncorrected SPL values are adjusted by multiplying them with a scale factor and by adding a (possibly negative) offset, both of which are specific to the device or model:

$$L_p = S_{phone} \cdot L_{p,\text{phone}} + \Delta_{phone} \tag{6.6}$$

The S_{phone} factor and the Δ_{phone} offset are constant for all levels and frequencies. At least 2 calibration points are needed to apply this method, in which case the values of S_{phone} and Δ_{phone} are found by solving the linear equation of the line that connects both points. For instance, if the mobile app measures 35 and 68.5 dB(A) while the reference respectively indicates 50 and 70 dB(A), the line's slope is $S_{phone} \approx 0.597$ and its y-intercept is $\Delta_{phone} \approx 29.104$. Alternatively S_{phone} and Δ_{phone} can be found by means of linear regression across more than two calibration points. Either way, as shown by chart 6.3c the resulting graph is a straight line.

Apps that have a user-accessible calibration setting typically suggest that users calibrate at 50, 60 or 70 dB[96]. As shown by the charts, the choice of the calibration point(s) can significantly affect the results.

[96] As for the sound source, some suggest white or pink noise [589, 596], others do not specify anything.

The problem with these methods is that they assume the systematic error can be approximated by a linear function (i.e. a straight line). Looking at the calibration points for the Nokia 5230 we see the error varies from ± 1.5 to almost 16 dB, and that it first increases, then decreases, and then increases again. On devices of other brands/models which we have tested [376] the error is usually not constant either, nor linearly in-/decreasing across the level range[97]. Hence, approximating errors with 1 straight line over a range of up to 100 dB always produces less accurate results than our linear interpolation method. Moreover, at some levels these methods exaggerate rather than reduce errors[98]. However, our method requires more calibration points to be determined, which takes more time[99].

6.7 Conclusion

In this chapter we provided a detailed discussion of the functionality, design and implementation of NoiseTube Mobile. Moreover we gave an overview of related (research) projects and applications developed by others in- and outside academia.

NoiseTube Mobile is one of the main contributions of this thesis. The app allows anyone to turn their mobile phone into a reasonably accurate sound level meter. It supports measuring the equivalent continuous sound level, data logging, data submission via the Internet and geo-localisation, features that are only available on very expensive SLMs. As motivated in section 3.5 this participatory sensing app can be used as a tool for personal, group or mass sensing. In other words, users are in full control of where and when measurements are made, whether or not they are geo-tagged with GPS coordinates, and whether or not they are shared with others through the NoiseTube Community Memory in order to contribute to local noise mapping efforts. Moreover, users can enrich the measurements via social tagging. NoiseTube Mobile is currently available for the Java ME and Android platforms and will soon be released for iOS, at which point it will run on roughly 80-90 % of the smartphones sold in 2011 (see section E.2.2).

In developing, refactoring, testing and deploying this app we have sought to tackle three of the primary challenges identified in section 3.5.2. First and foremost we have helped to move mobile sensing research into practice (challenge 1). In the last two years alone NoiseTube Mobile has been downloaded over 12000 times and it is being used by citizens from all over the world. Secondly, we have ensured that the app can produce data

97 We should note that the HTC Hero, for which calibration points are shown in chart 6.2, is the best model we have come across so far, in terms of both the size and the linearity of the error.

98 For example, consider the red line in chart 6.3b.

99 In fact our linear interpolation algorithm requires just 1 calibration point – in which case it basically applies scaling along a line between the origin and that point. But for good results we advise to calibrate every 5 dB – in which case our algorithms basically applies scaling–trimming over intervals of 5 dB – from 30 to 100–105 dB(A), resulting in 15–16 points.

of acceptable quality (challenge 4). We did this by closely following the specifications for (integrating-averaging) SLMs, such that we are able to measure the A-weighted equivalent continuous sound level (L_{Aeq}), and by devising a calibration/correction method that is superior to those used in related work. In the next chapter we will discuss the calibration process and the evaluation of the achievable accuracy in more detail. Thirdly, we have contributed to the availability of reusable, open source components for mobile sensing (challenge 5). We did this by applying various software engineering best practices, including many design patterns, throughout the development and refactoring of the app. As a result we have facilitated variability and reuse on three dimensions:

- **Platform variability**
  The current two variants of NoiseTube Mobile – targeting Java ME and Android – are built on a cross-platform architecture in which a significant part of the source code is shared (i.e. reused), which makes it easier to maintain and co-evolve the apps in parallel. Moreover, the strict separation of core functionality and platform-specific code has simplified porting the software to the iOS platform.
- **Device variability**
  Despite standardised software platforms there is a lot of variation among the smart- and feature phones on the market today. To make our software robust we made sure it can deal with differences in hard- and software functionalities, platform component versions, bugs and limitations at the platform level, and, most importantly for our purposes, variations in the audio recording path.
- **Application variability**
  The architecture can be easily extended to support the collection, processing and storage/transmission of other types of sensor data besides SPL measurements. Moreover, the sound level measuring classes were designed to be nearly independent from rest of the NoiseTube system, and can thus be easily reused in other contexts.

Because NoiseTube Mobile is open source others can make contributions of their own – by verifying, improving and extending it – or benefit from our efforts – by reusing parts of it.

Our discussion of related work has made it clear that, while ours was not the first, nor the last, attempt at creating a mobile sensing system for sound level measuring and assessment of environmental noise, the combination of NoiseTube Mobile and the Community Memory is the most complete solution to date, and in all probability also the most widely used one. Moreover, the appearance of many similar projects and apps, some of which were likely (in)directly inspired by NoiseTube, confirms that our efforts were visionary and remain highly relevant.

In the next chapter we explain how we have evaluated and improved the NoiseTube system and associated strategies for collaboration and coordination, through extensive experimentation in the lab and in the field. In chapter 8 we discuss a number of opportunities for future improvements of the NoiseTube system.

Chapter 7

Validation

7.1 Introduction

In the previous two chapters we have introduced the NoiseTube system and the participatory approach to environmental noise assessment it enables. In line with our ambition to move mobile sensing into everyday practice (see challenge 1 in section 3.4), it is essential that we validate our proposed solution – in terms of suitability and usability – through experiments in laboratory *and* real-world conditions, which is the goal of this chapter.

The main hurdle that must be overcome for participatory sensing to be accepted as a suitable method for environmental monitoring has to do with *data quality* (see challenge 4). This means we should formulate an answer to the widely-held view, especially among governmental officials and institutions, that data collected through participatory sensing is neither precise (due to random errors) nor accurate (due to systematic errors) enough to inform policy decisions. There is a technical and a human side to the challenge of data quality. The technical side pertains to the performance (in terms of accuracy and precision) of the devices used and how the data is aggregated (e.g. to produce noise maps). The human side relates to the skill, dedication and compliance of those doing the actual fieldwork (i.e. measuring noise) and the way efforts are coordinated (challenge 3). It is certainly true that individual sensors (i.e. mobile phone microphones) are less accurate than equipment used by professionals, and that user behaviour may have detrimental effects on data quality as well. However we hypothesise that the former can be compensated by rigorous calibration (to reduce systematic errors), and that both can be compensated by statistical reasoning (to reduce random errors and increase representativeness) over large datasets with dense spatio-temporal granularity, which participatory mobile sensing allows to amass at low cost. To test this hypothesis, and thereby validate our approach, we conducted a series of laboratory and real-world experiments.

We first report on lab and field experiments in which we tested, improved (through calibration and averaging), and validated the performance of mobile phones as sound level meters, while taking applicable norms and regulations into account insofar as possible. This work served a twofold purpose. First, the calibration experiments were needed in preparation of later participatory noise mapping experiments (see below). Second, they allowed us to test and confirm a number of assumptions lying at the basis of how NoiseTube Mobile corrects measurements using calibration points (see chapter 6). As a result we can now provide insight in the accuracy one can expect to achieve, before and after calibration, for artificial sounds of different frequencies and amplitudes, as well as for real-world sounds in real-world conditions. Moreover we can now quantify how microphone characteristics vary among devices of the same model and how that affects calibrations. As noted in section 3.4, to tackle challenge 1 it is useful to forge partnerships with domain experts. Therefore these experiments were carried out in collaboration with professional acousticians from the VUB's Department of Mechanical Engineering.

To evaluate our participatory approach for the assessment of environmental noise (see section 5.3) in practice – in terms of usability, data quality and organisational aspects – we set up a two-phased noise mapping campaign which ran for 3 weeks in the city of Antwerp, in collaboration with a local grassroots environmental activism organisation – making it a realistic community science experiment. A total of 13 members volunteered to carry out measurements in an area of $\pm 1\,\mathrm{km}^2$. That allowed us to investigate the performance of the NoiseTube system in the hands of end-users, try out different coordination strategies, and to assess the validity and utility of the resulting participatory noise maps by means of qualitative comparison with official, simulation-based noise maps of the area. Because the adoption by the wider public is so important for participatory noise mapping to become successful, we also conducted a small user study to evaluate the NoiseTube system and our noise mapping approach from the users' perspective. This was done by means of a questionnaire filled out by the participants of the experiment in Antwerp. While the group is too small to draw general conclusions, the feedback gathered – regarding usability of the tool, participant motivations, etc. – is still extremely useful to fine-tune the software and future (experimental or operational) noise mapping campaigns. At the same time it was an opportunity to test the questionnaire as an evaluation tool for future use with other NoiseTube users.

The experiments to assess the performance of mobile phones as sound level meters and the findings regarding achievable accuracy and precision are reported on in section 7.2. Section 7.3 covers our participatory noise mapping experiment in Antwerp, discusses the pros and cons of participatory vs. simulation-based noise maps, and summarises the main findings of the user study. We conclude this chapter in section 7.4.

This chapter is based on an article [127] that is due to appear in the *Pervasive and Mobile Computing* journal.

7.2 Mobile phones as sound level meters

The first thing we need to evaluate is the equipment used to collect data in the field. In other words, this means we need to test, and if necessary improve – primarily through calibration – how mobile phones perform as sound level meters.

7.2.1 Requirements

If we want to compare our results to those of official assessments it is important to play by the same rules insofar as possible. As in chapter 4, we focus on the legislative situation in the EU. The EU's Environmental Noise Directive [173] (END), although written with simulation-based approaches in mind, allows strategic noise maps to be based on measured rather than calculated values [173: Annex II][1]. When authorities opt to use measurements they are bound by additional European [157] or national [488] norms and recommendations. Such regulations are primarily aimed at supporting strategic noise mapping policies (as required by the END and/or national legislation), which is apparent from the fact that separate guidelines are prescribed for different sources of noise. Hence measurements made in the field serve mainly as a basis for the calculation of more general (long-term) noise exposure values and input parameters for simulation models. As an example, box 7.1 summarises the equipment requirements for the measuring of road traffic noise, as prescribed by the Dutch government in its 2006 regulation on calculation and measuring of environmental noise [488], which is also used in Flanders [313: p. 7].

1. A-weighting;
2. direct read-out of sound levels in dB(A);
3. computation of "overall" L_{Aeq} values[2] over arbitrary time intervals;
4. calibrated with respect to microphone parameters;
5. microphone with omnidirectional sensitivity and a windscreen;
6. a wind direction meter;
7. a wind speed meter;
8. a device to measure the speed of passing vehicles.

Box 7.1: Official Dutch equipment requirements for the measuring of road traffic noise [488: Bijlage III, pp. 28–29]

[1] Refer to sections 4.3.2.7 and 4.3.2.8 for a discussion of the END strategic noise mapping policy and the conventional methods authorities apply to implement it.

[2] Refer to section A.5.1.3.2 for the definition of L_{Aeq} and the calculation of "overall" L_{Aeq} values.

Requirements 1 to 5 correspond to common features of integrating-averaging sound level meters[3]. The Dutch regulations [488: Bijlage III, p. 29], as well as those in many other countries, require measurements to be made with Class 1 SLMs [268]. In order to compute generalised exposure values or parameters for simulation models, the prescribed calculation methods account for local meteorological and traffic conditions, hence the need for requirements 6 to 8.

As explained in chapter 3, participatory mobile sensing assumes that citizens collect data using multi-purpose, off-the-shelf devices – typically mobile phones. Moreover, participation in a sensing campaign should not render "normal", everyday usage of the device impossible[4]. For this reason requirements 5 to 8 are incompatible with our approach. NoiseTube users should be able to carry out measurements where- and whenever they want – typically during their everyday journeys – using only an off-the-shelf phone – ideally their own. Hence we cannot expect them to carry (let alone purchase) additional devices, nor ask them to modify their phone (e.g. cover it with a windscreen).

Requirements 1 and 2 are satisfied by NoiseTube Mobile, which displays series of $L_{\mathrm{Aeq,1s}}$ measurements and can log them to a file. The acrapp also displays the overall L_{Aeq} for the running track and alternatively this can be computed from logged $L_{\mathrm{Aeq,1s}}$ series, possibly from multiple subsequent tracks. Hence we can comply to requirement 3. This leaves us with requirement 4, to which we pay special attention here.

7.2.2 Equipment selection

For the experiments discussed here we have used a set of 11 Nokia 5230 [386] handsets. The Nokia 5230 is a fairly cheap feature phone[5] but is nevertheless able to run NoiseTube Mobile (Java ME variant). To compare devices we have given each a number from 0 to 10. Henceforward we refer to a particular one as "Nokia 5230 #*i*" or just "phone #*i*".

While focusing on a particular model, we have not acquired all 11 phones in one batch. Instead, four of them (#0 to #3) were bought new and the rest were bought second hand. Phone #0 was bought in April 2010 and #1, #2 and #3 were bought together in June 2010, all 4 at the same shop in Brussels. Phone #1, #2 and #3 presumably originate from the same manufacturing lot. The second hand phones were bought separately between July and November 2010, through eBay and similar websites from individual sellers in Belgium and in Germany – hence these have different dates of manufacture and diverse usage histories.

[3] Refer to section A.5.1 for a thorough introduction to sound level meters (SLMs).

[4] As discussed in section 3.3.4, this concern is referred to as *transparency* in mobile sensing literature.

[5] Refer to section E.1 for an explanation of the vague difference between smart- and feature phones.

The main reason for working with this heterogeneous set of mostly second hand devices is that we consider it a realistic example of what a citizen's organisation might put together to set up a participatory noise mapping campaign. Moreover it allows us to test the assumption that the microphone characteristics of different instances of the same phone model are sufficiently similar to correct measurements made on one instance using calibration settings determined on another (see below). Getting second hand phones also meant we could lower costs – the new phones cost €150 a piece[6] while we paid only €98 on average for the used ones – which was a bonus.

7.2.3 Calibration

As explained in appendix B, calibration is a procedure to detect and correct systematic errors of a measuring device by comparing its measurements with those of a trusted reference device. In our case, we must compare the SPL measurements made with NoiseTube Mobile on mobile phones, with those made (at the same time and place) by a professional, calibrated SLM – i.e. the reference – which we consider to be correct (i.e. true). By exposing different phones and the reference to sounds of varying amplitudes (i.e. sound levels) and possibly frequencies we can determine the microphone characteristics of each phone, which boils down to an estimation of the systematic error at different amplitudes and frequencies. This information, which we have called "calibration points" in chapter 6, can then be used to correct for such errors later on.

Below we discuss a series of calibration experiments which have served a dual purpose. On the one hand they were conducted in preparation of the real-world participatory noise mapping experiments discussed in section 7.3, for which we used the same set of (then calibrated) devices. On the other hand, as announced in chapter 6, we needed to test a number of assumptions lying at the basis of how NoiseTube Mobile corrects measurements using calibration points (see pages 184 to 187).

This is reflected in our choice to calibrate each of our 11 Nokia 5230s individually, even though they are of the same model. In preparation of our real-world experiments – intended to evaluate the full potential of participatory noise mapping including the accuracy of the resulting maps – this makes sense because it allows us to maximise accuracy at the level of individual devices[7]. Moreover, as mentioned above, comparison of experiment results for the 11 phones lets us test the assumption that instances of the same model behave sufficiently similar to warrant, for general purposes, the correction of measurements using calibration points determined for the model[8], rather than the individual device.

[6] The Nokia 5230 was recently discontinued, the last retail price was as low as €69.

[7] This means we take control of an otherwise ignored parameter (i.e. device-specific systematic errors).

[8] As in the publicly available version of NoiseTube Mobile, which choses a calibration setting by detecting the brand and model of the phone on which it runs (see pages 184 to 187).

A second assumption, mentioned in chapter 6, is that we can achieve sufficiently accurate results without considering the frequency domain[9]. In principle information about the microphone's frequency response could be used to design a digital filter that would (if added to NoiseTube Mobile) correct[10] measurements through adjustments depending on both frequency and level. While this approach could produce more accurate results than our current solution[9] there are some important downsides – which we list below – to using a digital filter. In other words, the second assumption boils down to the suspicion that a digital filter is just not worth the trouble. To test whether this is valid we first subjected two phones to *pure tone* calibration experiments, giving us insight in their frequency responses, before moving on to *white noise* calibration for all 11 phones.

A third, implicit assumption in the way we handle calibration and correction is that microphone characteristics remain stable over time, or at least do not change enough to necessitate *re*calibration, such as is done with professional SLMs (see section A.5.1.1). We tested this assumption by repeating a white noise calibration experiment on the same phone after 5 months.

7.2.3.1 Setup & procedure

Calibrating phones requires a controlled, quiet environment to carry out sound level measurements, a signal generator, and a calibrated SLM to be used as a reference.

We have been able to conduct our calibration experiments in the best possible conditions by using the acoustic lab facilities of the VUB's Department of Mechanical Engineering. There we have used an anechoic chamber[11], a HP Agilent 33120A [8] waveform generator to produce pure tone signals – sounds consisting of a single frequency – and a Brüel&Kjær Type 1405 noise generator to produce white noise. The signals were amplified using a Brüel&Kjær Type 2706 amplifier and then fed to a loudspeaker to produce the actual sound in the anechoic chamber. The reference sound level meter setup consisted of a Microtech MK250 high-end condenser microphone [347], an LMS Scadas III data acquisition station [322], and a PC running the LMS Test.Lab software [323]. The signal generators, LMS station and PC were positioned outside the anechoic chamber, while the amplifier, loudspeaker, reference microphone and the mobile phones were positioned inside it. Obviously we also made sure that the reference microphone and those of the phones were directed at the loudspeaker and positioned at the same distance from it.

On the phones we ran the NoiseTube Phone Tester (Java ME variant) with sound being recorded at 48 kHz, the highest supported sampling rate for the Nokia 5230. As explained

[9] The correction algorithm in NoiseTube Mobile (see pages 184 to 187), only considers the measured sound level and uses calibration points indicating systematic errors at different levels (not frequencies).

[10] Much like an A-weighting filter corrects for the frequency response of human hearing.

[11] A sound-proof, echo-free room, ideal for calibration experiments.

in chapter 6 the NoiseTube Phone Tester acrapp was derived from NoiseTube Mobile for testing and calibration purposes. It uses the exact same sound level measuring code but does not apply any corrections to the measured values.

The general procedure we followed in our experiments went as follows. We produced a sound of a particular type (pure tone of a certain frequency, or white noise) and constant level over an interval of about 1 minute, during which the sound level measured by the reference was read directly from the PC (and written down), and the values measured on each tested phone were time-stamped and logged to a file by the Phone Tester app[12]. When the interval was over the sound level and/or frequency was changed and then again kept constant for ± 1 minute, until all relevant levels and/or frequencies were tested. Afterwards the log file from each phone was analysed. Using the timestamps we identified the measurements corresponding to each interval and then averaged them to eliminate random errors, resulting in a single SPL value measured by that phone for that type and level of sound. These averaged values were then compared with the corresponding values measured by the reference in order to compute (systematic) measuring errors.

We should note that in absence of an anechoic chamber and dedicated signal generation and SLM hardware this procedure can also be carried out in a quiet room (preferably with little echo), using a PC to generate the signals[13], a loudspeaker (preferably amplified and of high quality) and a portable SLM device to use as a reference. However, the less ideal environment and less accurate equipment may affect the accuracy of the results.

7.2.3.2 Pure tone calibration experiments

To investigate the frequency response of their microphones we subjected two[14] of our phones (#0 and #1), as well as the reference microphone, to signals with frequencies spanning about the entire human hearing range (see section A.3.2).

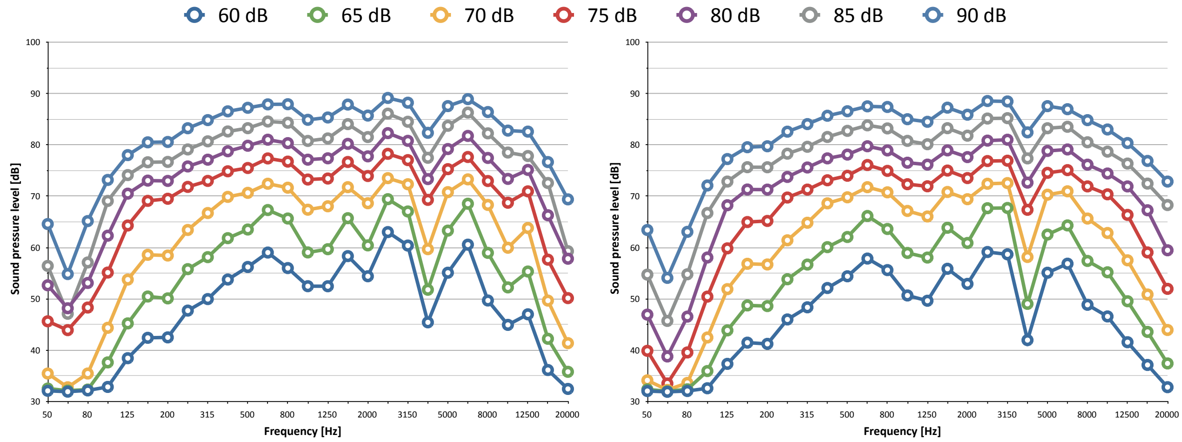
Concretely, we produced pure tones at every third octave band from 50 Hz to 20 kHz, for a total of 27 tones. This sequence was traversed a total of 7 times, for sound levels from 60 to 90 dB with intervals of 5 dB. Each sequence was started at 1 kHz with the amplifier adjusted such that the reference measured exactly that level (e.g. 65 dB). Then, the frequency was adjusted – down to 50 Hz and up to 20 kHz – while the amplifier was left untouched. At each frequency, which was kept constant for about 1 minute, the

[12] Remember that NoiseTube Mobile and the Phone Tester create series of $L_{\mathrm{Aeq,1s}}$ measurements, and that on the Java ME variants there is a gap of about 1 s between subsequent measurements. Hence, in the log files we get ± 30 $L_{\mathrm{Aeq,1s}}$ measurements per minute.

[13] For instance with free software like Audacity [34].

[14] When we conducted this experiment we had only 4 phones (the 2nd hand ones had not been bought yet). Because we were uncertain about whether pure tone calibration was at all necessary, we started with just 2 – enough to compare device-specific characteristics.

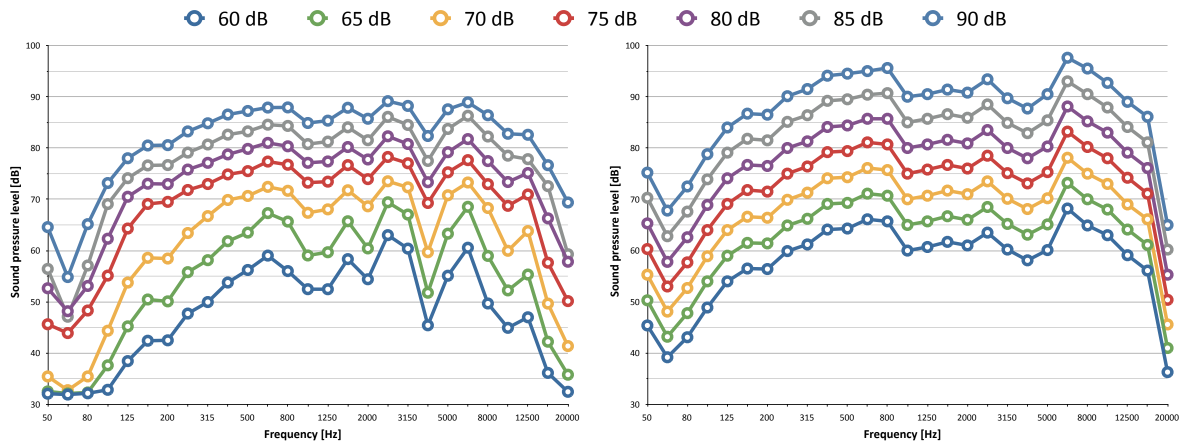
level measured by the reference level was written down (while the measurements made on the phone were logged constantly). This experiment resulted in 189 (= 27 × 7) SPL measurements per device. Chart 7.1 shows the results for phones #0 and #1.



(a) Nokia 5230 #0 **(b)** Nokia 5230 #1

Chart 7.1: Pure tone calibration results for Nokia 5230 #0 and #1

Visual and numerical comparison of the results for phones #0 and #1 led us to conclude there are no significant frequency-dependent differences between both devices. Hence for comparison with the reference microphone we focus on the characteristics of one phone. Chart 7.2 shows the results for phone #0 and the reference.



(a) Nokia 5230 #0 **(b)** Reference

Chart 7.2: Pure tone calibration results for Nokia 5230 #0 and the reference

Looking at chart 7.2b the first thing we notice is that, contrary to what one might expect, the graph lines are not flat. In theory, if the reference microphone had perfect frequency response we should get a straight horizontal graph line for each sound level in chart 7.2b. However, besides the fact that truly "perfect" microphones do not exist, the results are likely also caused by imperfections in other parts of the setup. One suspect being the loudspeaker – as speakers with "perfect" reproduction at all frequencies do not exist either.

Although we were initially surprised by these results, acousticians assured us that they were to be expected and that the reference could be used to calibrate our phones against.

By comparing charts 7.2a and 7.2b we can clearly see a difference in spread between subsequent sequences for the phone, while those of the reference are spaced equally (±5 dB apart). This tells us that there is a significant dependence on amplitude, or, in other words, that the microphone of the phone is not equally sensitive at all sound levels, ruling out correction by a constant offset (see appendix B).

By comparing the form (rather than the height) of corresponding lines in charts 7.2a and 7.2b we see that the frequency responses of the phone are quite similar to those of the reference, although the variations are more exaggerated (e.g. the dip at 4 kHz). The main difference occurs at frequencies below 100 Hz and levels below 75 dB. A smaller but noticeable deviation happens at 12.5 kHz, where the phone exhibits a peak that is not present in those of the reference.

These results indicate that in all likelihood the Nokia 5230 (fortunately) does not contain signal processing hard-/software[15] to amplify or dampen certain frequencies. Moreover, the results show that sound levels have a more significant impact on systematic errors than frequencies do. We therefore concluded (together with acousticians) that, while far from perfect, the frequency responses of Nokia 5230 #0 and #1 are probably "flat enough" in order to achieve sufficiently accurate results with level-dependent correction alone. In other words, designing a digital filter would not be necessary.

This is good news because there are several arguments against digital filters, especially when looking beyond our validation work and considering what would be feasible for others to repeat on their own. For one thing, designing[16] such a filter is a complex undertaking that requires knowledge of digital signal processing, which we cannot expect our end-users to have. Although it may be possible to automate that process[17], the calibration experiment itself takes a considerable amount of time and dedication. Not counting the time to set up equipment beforehand and to analyse log files afterwards, it takes at least 189 minutes, whereas the white noise calibration experiments discussed below only take about 16 minutes. Since (white noise) calibration is already a bottleneck to adoption of participatory noise mapping, we should be careful not to make it even more complicated – and out of reach of average citizens and their organisations – without good reason. Another argument is that correcting measurements using such a filter would be more computationally intensive than the current linear interpolation algorithm. While today's smartphones would probably have no trouble with the extra workload it would nevertheless (slightly) increase the power consumption.

[15] Such as a low-pass filter, as warned about by Santini et al. [459].

[16] This boils down to determining the coefficients for a FIR or IIR filter [262] based on the results of pure tone calibration experiments.

[17] Such that end-users could compute coefficients using a desktop application or Web service.

7.2.3.3 White noise calibration experiments

Having decided not to pursue the development of digital filters for the correction of systematic measurement errors, it was not necessary to submit the other 9 phones to a pure tone calibration experiment and the level-dependent correction algorithm implemented in NoiseTube Mobile could be left unchanged. However, this left us with the choice of a type of sound to calibrate against – i.e. to determine the "calibration points".

The frequency responses for phone #0 and #1 indicate that it would be a bad idea to calibrate against a single pure tone[18], since that might introduce a bias for that frequency. Instead we should account for multiple frequencies at once. The standard way to do that is to calibrate against an artificially generated signal composed of a fixed mix of all frequencies in the range considered. The most common options are *white* and *pink noise* [589, 596]. The difference between both is that in white noise all frequencies have equal intensity, whereas in pink noise intensity is inversely proportional to the frequency. Pink noise is often used when there is a focus on low frequency sounds. Following the advice of acousticians we decided to calibrate against white noise signals with a frequency spectrum up to 20 kHz, in order to avoid any frequency bias.

The procedure we followed is similar as above. Instead of exposing the 11 phones (and the reference) to pure tones at different frequencies and different levels, we used white noise at 16 different sound levels, from 30 to 105 dB(A) (as measured by the reference) in steps of 5 dB, for about 1 minute per level. Afterwards the measurements made on the phones during each 1 minute interval were averaged (to eliminate random errors), resulting in 16 SPL measurements per device. Chart 7.3 shows the results for each phone.

Each of the dots on this chart represents a calibration point: a couple of SPL values, one measured by the reference[19], the other being the average of the measurements made by the NoiseTube Phone Tester on the phone in question[20]. We should note that the variability before averaging was very low (standard deviation < 0.15 dB, for phone #1). As explained in chapter 6, NoiseTube Mobile uses such points to apply level-dependent adjustments to compensate for systematic errors, with each adjustment being calculated by linear interpolation between a suitable pair of points. As argued in section 6.6.2 this method is superior to the trimming, scaling and scaling–trimming used in related work, namely because the calibration points do not form a straight line, as is clear from chart 7.3.

Chart 7.3 also provides insight in the variability among the 11 Nokia 5230s. Overall, they behave very similarly, with the notable exception of phone #9 which is further off.

[18] Although that is what happens when a professional SLM is calibrated using a "slide-on" calibrator device, which typically produces a pure 1 kHz tone at a fixed sound level (see section A.5.1.1).

[19] We have called this value the *output*.

[20] We have called this value the *input*.

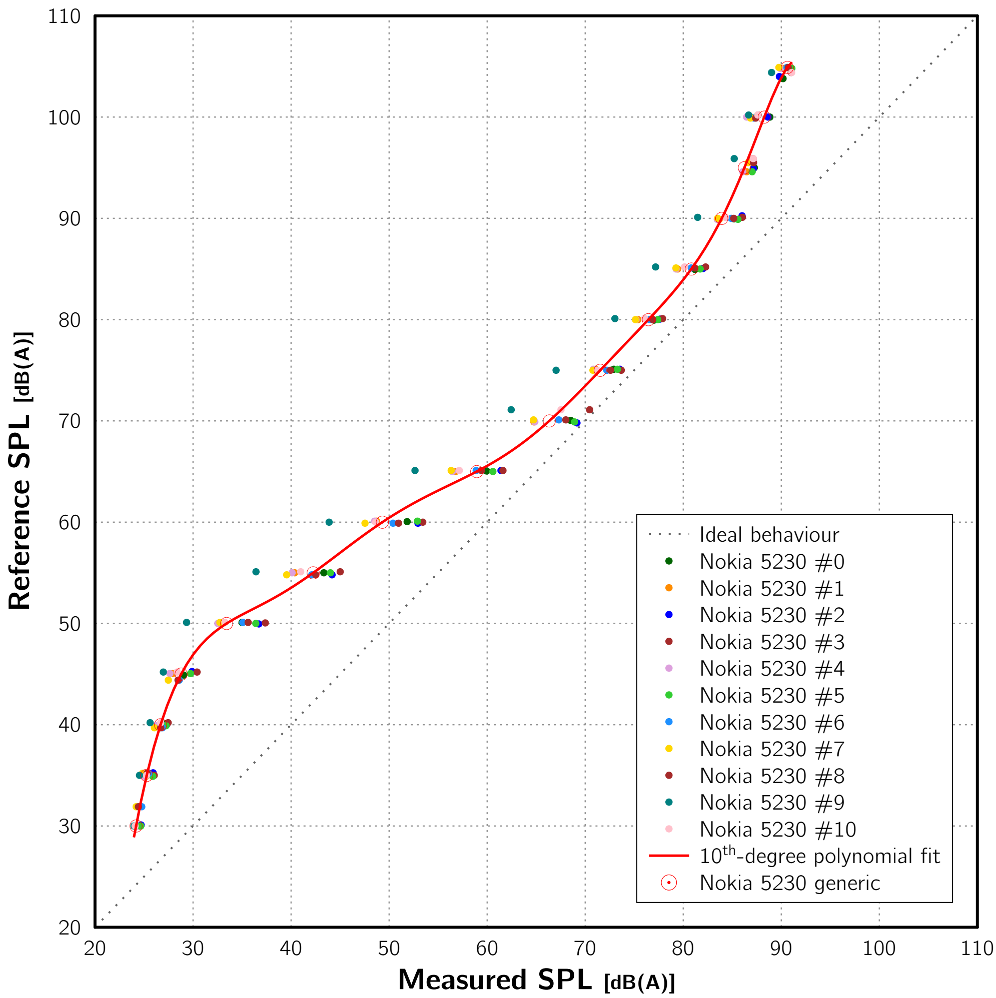


Chart 7.3: Calibration points for our 11 Nokia 5230 handsets (based on white noise calibration), plus the "average" calibration points for a generic Nokia 5230

Still our assumption that devices of the same model have similar microphone characteristics seems to largely hold. However, to know to what extent accuracy would be affected if measurements made on one phone are corrected using calibration points determined on another, we simulated the outcome of every combination. The procedure went as follows. We took the calibration points C_x of phone x and the calibration points C_y of phone y. For every point $(c_{x,i}^{\mathrm{IN}}, c_{x,i}^{\mathrm{OUT}})$ from C_x, we took the input value $c_{x,i}^{\mathrm{IN}}$ (an uncorrected measurement made on phone x) and applied our correction algorithm[21] to it, using C_y as the calibration. Then we computed the absolute difference of the corrected value and the corresponding output value $c_{x,i}^{\mathrm{OUT}}$ (measured by the reference when phone x measured $c_{x,i}^{\mathrm{IN}}$). Then the 16 absolute differences were averaged and we moved on to the next combination of phones. Table 7.1 lists the results of this simulated "cross-calibration". For comparison we also included the average absolute difference (w.r.t. the reference) for the calibration points of each a phone[22] – indicative for measurement accuracy *without* calibration.

[21] As implemented in NoiseTube Mobile (see pages 184 to 187).

[22] For phone x with calibration points C_x this is calculated as $\frac{1}{16}\sum_{i=0}^{15}|c_{x,i}^{\mathrm{OUT}} - c_{x,i}^{\mathrm{IN}}|$.

Legend: error < 2; 2 ≤ error < 4; 4 ≤ error < 6; error ≥ 6

		Measurements made on Nokia 5230…												Average #0-#10
		#0	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	Generic	
No correction		8.1881	9.9356	7.6563	7.5394	10.1544	7.8375	8.9194	10.3863	8.6925	12.3300	9.5738	9.1780	9.2012
Corrected using calibration points determined for Nokia 5230…	#0		2.1519	0.7149	1.2196	2.5882	0.7944	0.9701	2.7894	0.9485	4.8764	1.7806	1.2194	1.8834
	#1	2.2577		2.7486	2.8324	0.6846	2.5479	1.4769	0.7076	1.6388	2.8042	0.7576	0.8361	1.8456
	#2	0.7286	2.7548		0.7282	3.1663	0.6771	1.6195	3.2721	1.5278	5.4050	2.4073	1.9287	2.2287
	#3	1.0332	2.8843	0.5076		3.4071	0.5756	1.6250	3.4089	1.4796	5.5188	2.0663	2.0317	2.2506
	#4	2.6381	0.6280	3.1412	3.2242		2.9355	1.8544	0.6865	1.9760	2.3320	1.0964	1.1973	2.0512
	#5	0.7223	2.5689	0.4668	0.4321	2.9992		1.2555	3.1344	1.0693	5.1762	1.7730	1.6466	1.9598
	#6	1.0924	1.5576	1.4330	1.4525	1.9615	1.0983		2.0023	0.5934	4.1097	1.0016	0.7151	1.6302
	#7	2.8124	0.5836	3.2970	3.3817	0.6580	3.0940	2.0153		2.0814	2.4403	1.2823	1.3416	2.1646
	#8	0.8155	1.7127	1.2438	1.3475	2.1212	0.9655	0.6215	2.0838		4.2780	1.0607	0.8372	1.6250
	#9	4.9644	2.8124	5.5167	5.5881	2.4510	5.2851	4.1544	2.2047	4.1681		3.2892	3.6115	4.0434
	#10	1.6257	0.7428	2.1661	2.1342	1.1275	1.9663	1.1131	1.2747	1.1273	3.3140		0.5952	1.6592
	Generic	1.2791	1.0024	1.8707	2.0713	1.3046	1.7931	0.7591	1.5389	0.8939	3.5917	0.7009		1.5278

Table 7.1: Average absolute errors w.r.t. the reference, over 16 white noise levels (30–105 dB(A)), without calibration and after simulated cross-calibration

Each value in this table is the average absolute error (w.r.t. to the reference) one can expect when measuring white noise between 30 and 105 dB(A), on the phone indicated on top, either without correction or after correction using the calibration points determined for the phone indicated on the left. The values are colour coded to highlight the size of the error. To interpret the values it may also help to consider that 3 dB is the smallest difference perceptible to the human ear[23]. Looking at the top row we see that without correction the average absolute error varies from 7.5 (for phone #3) to 12.2 dB (for #9). To study the effect of cross-calibration we now take a look at the 110 combinations of 2 *different* phones (#0–#10). At first glance the colour coding already reveals that, regardless of the chosen calibration points, the errors drop significantly after correction. In just over half ($\frac{57}{110}$) of the cases the error is < 2 dB, which is very small. For $\pm$ 37 % ($\frac{41}{110}$) of combinations it is between 2 and 4 dB. In the remaining 11 % ($\frac{12}{110}$) of cases, all of which unsurprisingly involve phone #9, the error is between 4 and 5.6 dB, which may be too big for some purposes, but is still significantly better than without correction.

Based on these results we can conclude that, in about 89 % of the cases, in which one of our Nokia 5230s is calibrated with points determined for one of the others, the average absolute error stays below 4 dB, which we consider acceptable for general usage[24]. Hence our assumption that it is sufficient to calibrate per model – rather than per individual device – can be considered correct in a vast majority of cases, at least for the phones we tested. While plausible we do not know whether the variability in our set of 11 Nokia 5230s is representative of that among devices of other models and brands.

If one can test only a single instance of a particular model, it is of course impossible to know to what extent it is representative for its peers. In other words, one cannot know

[23] See table A.3 on page 279.

[24] If we ignore the cases involving phone #9, 100 % of the combinations result in an error below 4 dB.

whether the device is an outlier, such as Nokia 5230 #9, whose calibration points could produce poor results on other instances. However, if one has calibration points for multiple instances of the same model, it is not only possible to compare device characteristics, but also to compute an "average" set of calibration points. Because reference levels in different experiment runs may differ, it is not always possible to just average the levels measured by each phone per reference level. Instead one should use polynomial regression to find a function the provides the best fit to all calibration points. By sampling the function at regular intervals one then obtains a new set of calibration points which is representative of the tested devices. Because this set encompasses information about multiple instances of the model in question, it can be considered an approximation of the behaviour of a random or "generic" instance of that model. Hence, if one has to pick a single set of calibration points to correct measurements on all instances of a particular model[25], such a "generic" set is a better choice than one determined for an individual instance.

For our 11 Nokia 5230s, or rather their 176 calibration points (11×16), we found that a 10th degree polynomial regression resulted in the best fit[26]. In chart 7.3 the resulting function is shown as a red line and the new calibration points as red circles. In the bottom row of table 7.1 we see the result of using these "Nokia 5230 generic" calibration points to correct measurements made on our 11 actual devices[27]. In the last column of table 7.1 we see that, on average, this indeed produces a better result that any of the calibration point sets determined for individual instances. Therefore the calibrations file that ships with NoiseTube Mobile[28] now includes these generic calibration points for the Nokia 5230, instead of a set determined for one instance [376].

Note that for our participatory noise mapping experiments, discussed in section 7.3, each of the phones was calibrated with its own calibration points, in order to maximise accuracy. The simulation approach described above cannot provide a meaningful estimate of the average error after such a device-specific calibration[29]. Therefore we conducted additional validation experiments in the lab and the field, which are discussed in the next sections.

In order to find out whether the microphone characteristics of mobile phones remain stable over time – another assumption we needed to test – we have submitted one of our phones (#1) to 2 white noise calibration experiments, in June and November 2010. During the intervening 5 months the device was regularly used in- and outdoors. As expected the difference in the performance of the phone was negligible: the average absolute difference

[25] This is necessary when using the publicly available version of NoiseTube Mobile, which only deals with model- or brand-specific calibrations, not device-specific ones.

[26] Coefficient of determination (R^2) = 0.9933; adjusted coefficient of determination ($\bar{R}^2$) = 0.9929.

[27] In the second-to-last column of table 7.1 we see the error one can expect when measurements made on a hypothetical random Nokia 5230 are corrected using calibration points determined for the individual instances. These values are not considered for the averages in the last column.

[28] And which the app automatically downloads from the NoiseTube website when it can access the Internet.

[29] If phone x = phone y the input values of C_x would be corrected using C_x as the calibration, resulting in an unrealistic average absolute error of 0 – which is why we left the diagonal cells in table 7.1 blank.

between corresponding measurements – after correcting for slight differences in the reference levels – was just 0.29 dB. Hence this phone did not require recalibration. Of course, we cannot draw firm conclusions based on experience with a single device. Moreover, 5 months may be too short and the way the phone was used in the meantime (mainly for noise mapping experiments) is probably not representative of typical mobile phone usage. Therefore additional testing is required to answer this question with more confidence.

7.2.4 Validation in the lab

To verify whether white noise calibration works as expected we set up a validation experiment in the anechoic chamber. The goal was the assess which systematic errors remain after (device-specific) calibration. We did this experiment with Nokia 5230 #0, on which we ran NoiseTube Mobile, configured to correct measurements using the calibration points for that particular phone. We followed the same procedure as before: the phone and the reference setup were exposed to 16 different levels of white noise, for about a minute per level, afterwards the measurements made by NoiseTube Mobile were averaged per interval[30]. This results in 16 points of comparison, as shown in chart 7.4.

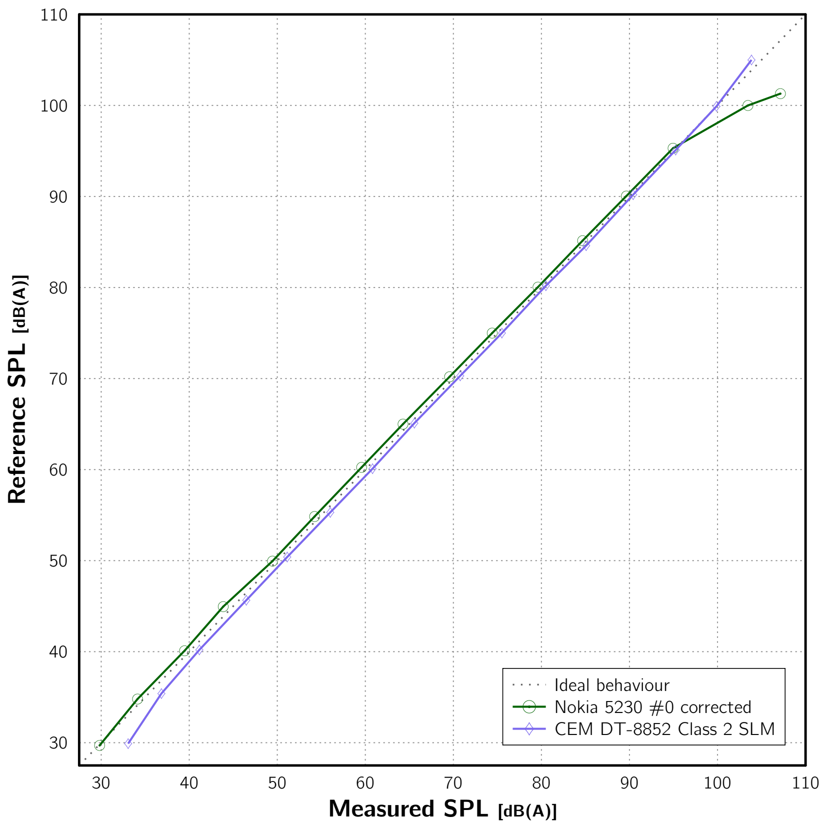


Chart 7.4: Measurements of white noise made by the calibrated Nokia 5230 #0 and the CEM DT-8852 SLM, in function of those made by the reference

[30] Again the variability before averaging was low (standard deviation < 0.38 dB).

We see that the measurements made by the phone now almost perfectly match those of the reference. Over the 16 comparison points we get an average absolute error of just 1.07 dB, and for those below 100 dB(A) that is just 0.56 dB. Considering that 3 dB is the smallest audible SPL difference, these average errors are very small indeed. However, at 100 dB(A) and above the error is more significant, indicating that the microphone of the Nokia 5230 becomes saturated – something we need to keep in mind when assessing very loud sounds. For comparison we also tested a calibrated Class 2 [268] SLM, namely a CEM DT-8852 [473]. As shown in chart 7.4, below 100 dB(A) the calibrated Nokia 5230 #0 performs as good, or even slightly better, than the SLM. Yet, as we discuss below, that does *not* mean NoiseTube Mobile meets the requirements for Class 2 SLMs.

In chart 7.5 we see the measuring error at different reference levels as they occurred in the validation experiment with Nokia 5230 #0. Also shown are the *uncorrected* result of the experiment – which we computed from the corrected one by simply applying the correction algorithm in reverse[31] – and the results of some cross-calibration simulations, in which the measurements made on phone #0 were (re)corrected using the calibration points for 4 other Nokia 5230s and those for a generic one. The chart makes it clear that while device-specific calibration gives in the most accurate result, model-specific calibration based on multiple instances (i.e. Nokia 5230 generic) performs almost as good.

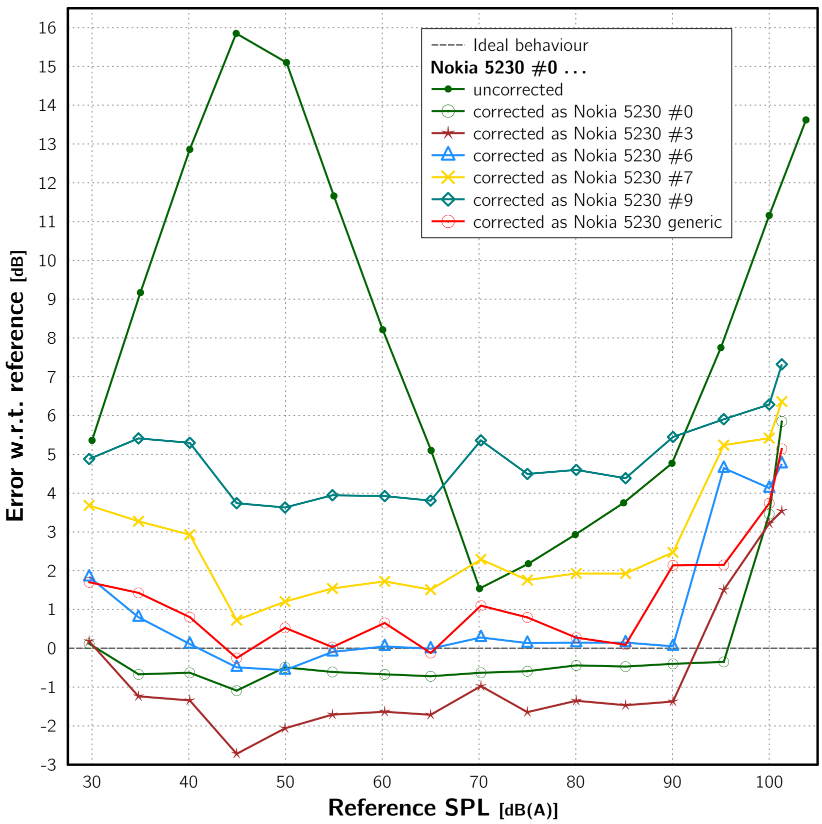


Chart 7.5: Measuring error of phone #0 (w.r.t. the reference) at different levels of white noise, before and after device- or model-specific white noise calibration

[31] By using input values as output values and vice-versa.

Although the outcome of the validation experiment is certainly satisfying, we must stress it was carried out in a controlled environment – free of external influences such as wind – using an artificial sound source (i.e. white noise), neither of which is representative of the situations in which NoiseTube Mobile is used in reality. Therefore, we also conducted an outdoor validation experiment, which is discussed in the next section.

While not representative of real-world conditions either, it is interesting to evaluate how NoiseTube Mobile, with a device-specific calibration based on white noise, would perform when measuring pure tones of different frequencies and levels. Therefore we have simulated the outcome of such an experiment. We took the results of the pure tone calibration experiment performed on phone #0 and fed the 189 (27 tones $\times$ 7 levels) measurements to the correction algorithm as it is implemented in NoiseTube Mobile, using the calibration points for phone #0. The resulting 189 values tell us what would have been measured if the phone had been running the calibrated NoiseTube Mobile app. With this simulated data it is possible to investigate how white noise calibration affects the systematic error at different frequencies and levels. This is illustrated by chart 7.6, which shows the errors (w.r.t. the reference) at different frequencies and levels of 60, 75 and 90 dB (as measured by the reference at 1 kHz). Chart 7.6a shows the original, uncorrected results[32], whereas chart 7.6b shows the simulated, corrected ones. Additional statistics, also for the 65, 70, 80 and 85 dB sequences, are provided in table 7.2.

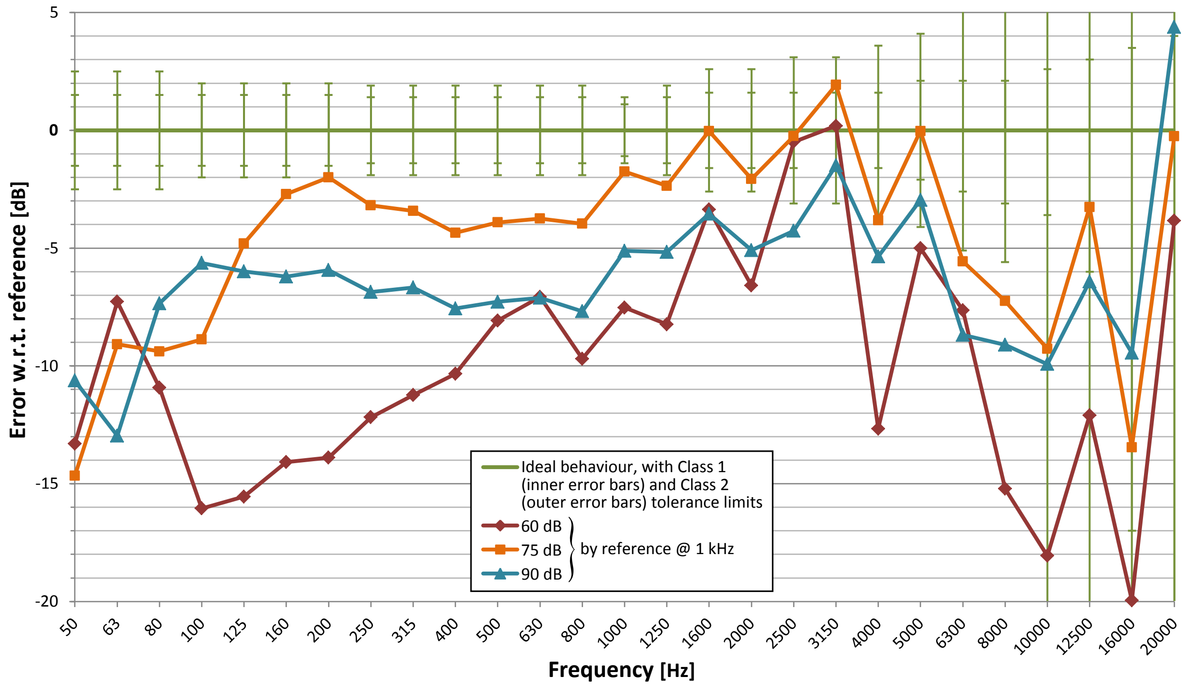
		Reference level @ 1 kHz						
		60 dB	65 dB	70 dB	75 dB	80 dB	85 dB	90 dB
Nokia 5230 #0 Uncorrected	Average absolute error	10.02	8.69	7.01	4.64	5.04	6.01	6.62
	Standard deviation of absolute error	5.04	5.24	5.60	3.90	3.05	3.48	2.47
	Maximum absolute error (@ frequency)	19.96 (16000 Hz)	18.87 (16000 Hz)	19.84 (50 Hz)	14.67 (50 Hz)	12.65 (50 Hz)	15.7 (63 Hz)	12.96 (63 Hz)
Nokia 5230 #0 Corrected	Average absolute error	3.18	2.98	2.99	2.51	2.54	3.00	3.45
	Standard deviation of absolute error	2.61	2.18	2.10	2.35	2.25	1.85	2.45
	Maximum absolute error (@ frequency)	11.58 (20000 Hz)	9.57 (20000 Hz)	8.25 (20000 Hz)	8.65 (20000 Hz)	8.44 (20000 Hz)	7.42 (50 Hz)	8.32 (3150 Hz)

Table 7.2: Statistics calculated over 27 frequencies for each of the 7 level sequences in the pure tone calibration experiment on Nokia 5230 #0, before and after simulated correction by means of device-specific white noise calibration

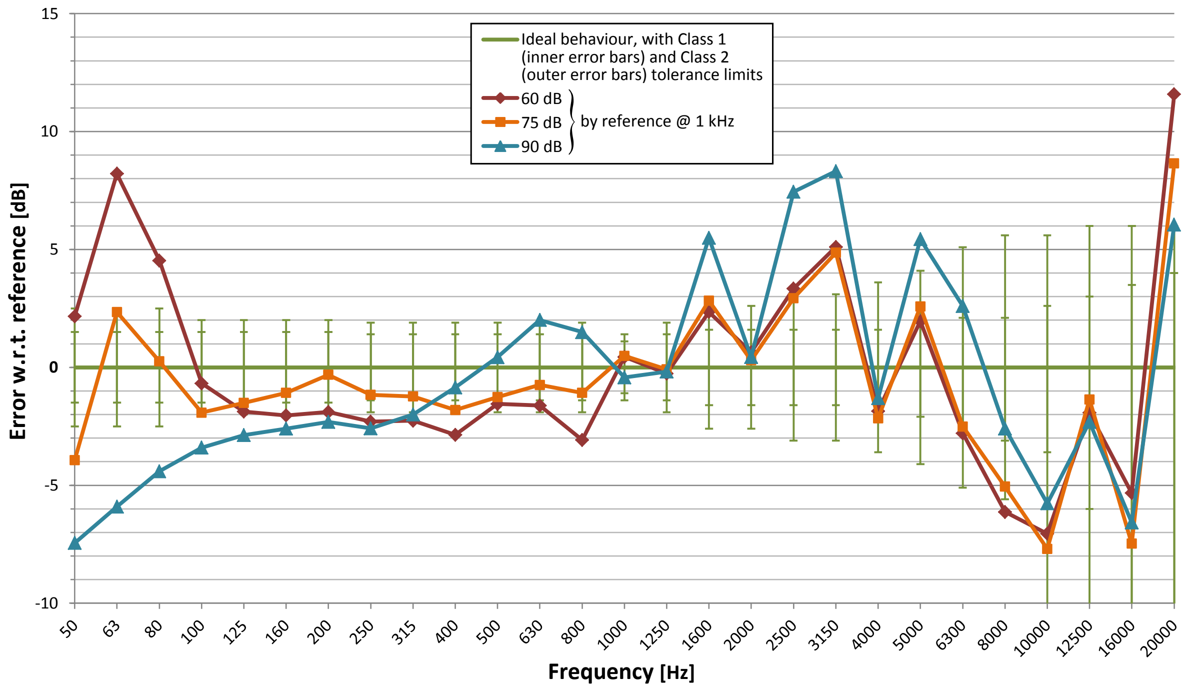
As shown by the chart and the table, both the size and the variability (standard deviation) of the errors are significantly lower after white noise calibration. However, the maximum absolute errors remain large and as expected they tend to occur either at very low or very high frequencies. To put these results into perspective, chart 7.6 also shows the error tolerance limits for Class 1 and 2 SLMs, as specified per frequency in the international standard IEC 61672-1:2002 [268: p. 16]. While NoiseTube Mobile does not meet the overall requirements for Class 2 SLMs[33], after device-specific white noise calibration the responses at 75 dB are almost completely within the limits. At 60 and 90 dB it also gets rather close, especially in a broad region in the middle of the frequency spectrum.

[32] This is just another view on results of the actual pure tone experiment, previously shown in chart 7.2.

[33] And consequently neither the stricter requirements for Class 1 SLMs.



(a) Nokia 5230 #0 uncorrected



(b) Nokia 5230 #0 corrected (simulated)

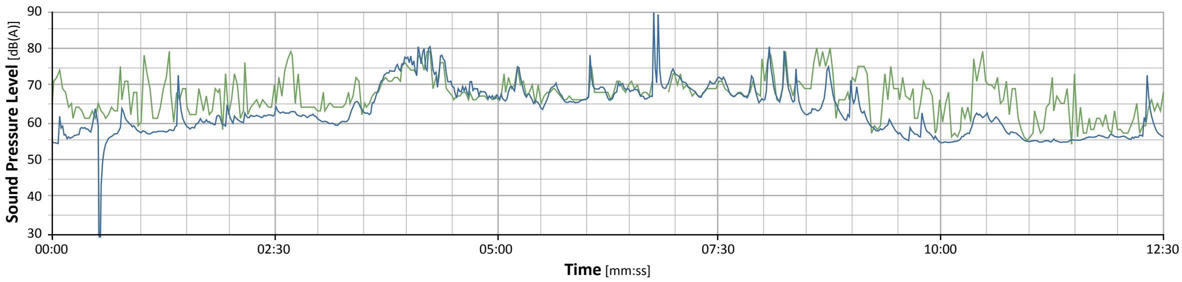
Chart 7.6: Measuring errors on Nokia 5230 #0 (w.r.t. the reference) for different frequencies along sequences of 60, 75 and 90 dB, before and after simulated correction by means of device-specific white noise calibration

Considering that the microphone we deal with is not at all designed for this purpose – nor is the rest of the phone's hardware – this is a surprisingly good result. We should also note that it is plausible that other (smart)phones on today's market have microphones with flatter frequency responses than the Nokia 5230, and would thus perform even better.

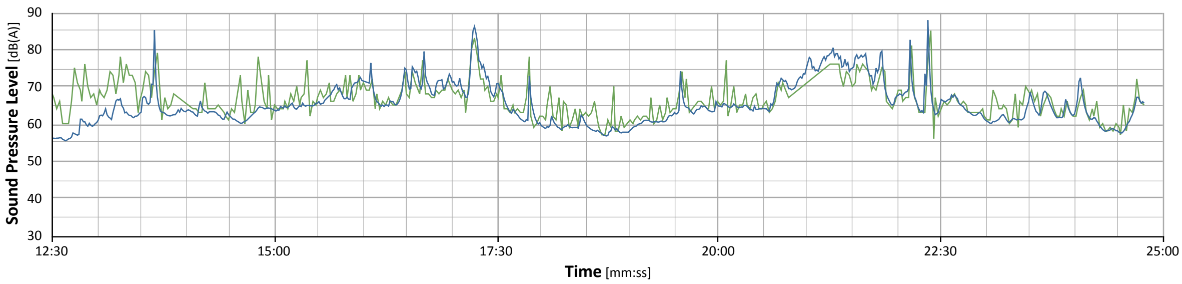
7.2.5 Validation in the field

When used for real noise mapping campaigns phones do not find themselves in an anechoic chamber with pure white noise coming in from one particular direction. Rather, they are exposed to the elements and to sounds, composed of very different and diverse mixtures of frequencies, coming from all directions. Consequently, in real-world conditions we should expect a lower accuracy compared to the results of our validation experiment in the lab (and to simulated outcomes based on lab results). To get an idea of the real-world performance of NoiseTube Mobile, with device-specific white noise calibration, we conducted a validation experiment in the field.

For this experiment we took a long walk in a (sub)urban area, while carrying the calibrated Nokia 5230 #0 as well as the CEM DT-8852 [473] SLM[34]. Both devices were held close together with the microphones facing forward[35]. The experiment took place on 2010-12-09 in the Linkeroever neighbourhood of Antwerp[36]. Chart 7.7 shows the sound levels, as measured by the phone and the SLM, during the first ±25 minutes of the walk.



(a) Part 1 (00:00–12:30)



(b) Part 2 (12:30–24:47)

Chart 7.7: A-weighted SPL measured by Nokia 5230 #0 (green) and a CEM DT-8852 (blue) during an outdoor walk

It is immediately apparent that the accuracy of the phone (w.r.t. the SLM) varies substantially. In some intervals there is a very good match (e.g. from minute 4 to 8), while in

[34] This Class 2 SLM advertises an accuracy of ±1.4 dB [473], but we found that – at least for white noise – it performs even better, with an average absolute error of just 0.77 dB (see chart 7.4).

[35] I.e. pointing away from the carrier and in the walking direction.

[36] The same area where the participatory noise mapping experiments, discussed in section 7.3, took place.

others the differences are much bigger (e.g. the first 3 minutes and from minute 9 to 13). While fairly moderate[37] the wind was an important factor here because, unlike the SLM, the phone did not have a windscreen. As the building density in the area varies and we regularly changed walking direction, exposure to the wind and the orientation of the microphones (w.r.t. to the dominant wind direction) changed regularly along the itinerary. Hence even if the speed and direction of the wind were constant its influence was not.

Another factor to take into consideration is that both devices were not measuring the same sound level *descriptor*[38]. As explained in chapter 6, NoiseTube Mobile acts as a so-called integrating-averaging SLM, measuring A-weighted *equivalent continuous sound level* (also called *time-average sound level*) over 1 s intervals, noted as $L_{\text{Aeq,1s}}$. In the Java ME variant, which was used in this experiment, there is a gap of ± 1 s between subsequent measurements, resulting in a rate of about 30 $L_{\text{Aeq,1s}}$ measurements per minute. On the other hand, the CEM DT-8852 is a so-called conventional SLM, measuring A-weighted *time-weighted sound level*, noted as $L_{\text{A}\tau}$, in which τ is a *time constant* that controls how fast the meter responds to sound level changes. For this experiment we set the time constant to 1 s, known as S(low) time weighting, and logged 1 measurement per second, resulting in a rate of 60 L_{AS} measurements per minute. While both sound level descriptors are measures of the same physical condition (i.e. SPL) there are inherent, slight differences in their values. Consequently comparison of individual corresponding $L_{\text{Aeq,1s}}$ and L_{AS} measurements is not a good way to quantify the accuracy of the phone[39]. Instead, we should compare logarithmic averages[40] for corresponding intervals.

Over the entire itinerary, which lasted ± 81 minutes, the calibrated Nokia 5230 #0 measured an average sound level of 71.05 dB(A), while the DT-8852 measured 70.90 dB(A). For the first ± 25 minutes, shown in chart 7.7, the averages are respectively 70.01 and 69.82 dB(A). Over the interval between minute 4 and 8, in which the phone matches the SLM very well (at least visually), we find average sound levels of 70.57 and 72.94 dB(A) respectively, which is not as good as for the longer intervals but still very close. For the windy interval between minute 9 and 13 the phone overestimated sound levels substantially, measuring an average of 68.34 dB(A) compared to 58.90 dB(A) on the SLM. This indicates that the lack of wind protection can cause temporary systematic errors of up to 10 dB. Although besides the wind, other factors may have played a role as well (e.g. different types of sound). Still, regardless of what caused the measurement errors, their effect all but disappears if we average the sound level over sufficiently long periods, as demonstrated by the negligible differences for the 25 and 81 minute intervals.

[37] On that day the average wind speed, maximum wind speed and maximum gust speed, as observed at the nearby Antwerp airport, were respectively 15, 28 and 35 km/h [565].

[38] Refer to section A.5.1.3 for a full explanation of the difference between the relevant sound level descriptors (including formulae) and the kinds of SLMs that measure them.

[39] In fact, even if both devices produced $L_{\text{Aeq,1s}}$ measurements at the same, consecutive rate, comparing measurements one by one would still be flawed due to inevitable (slight) timing differences.

[40] Also called "overall L_{eq}". Refer to section A.2.6 and section A.5.1.3.2.

7.2.6 Preliminary conclusions

Through calibration experiments in the lab and subsequent simulations we have studied 4 types of variability – devices, frequencies, amplitudes and times – in order to answer questions regarding the way calibration should be handled in general, and to prepare our devices for the real-world experiments discussed in the next section.

Through pure tone calibration experiments on two of our phones we found that the frequency responses of the Nokia 5230 are sufficiently flat for level-dependent corrections to give reasonably accurate results. Hence we have made a pragmatic decision not to develop digital filters (for frequency- and level-dependent corrections), as not avoid making the need to calibrate an even bigger bottleneck to adoption then it already is.

Next we submitted the 11 phones to white noise calibration experiments. In white noise all audible frequencies are present in equal proportions, allowing to calibrate devices in function of amplitudes alone, while avoiding a bias for particular frequencies. We calibrated each phone separately in order to reduce systematic errors on each phone to a minimum (in preparation for real-world experiments), and to assess the difference in response (to various levels of white noise) among devices of the same model. While outliers do occur we found that in general responses are very similar. Through simulations we then investigated the errors one can expect when correcting measurements on one device using calibration points determined on another of the same model. We found that, for combinations of distinct pairs from our set of 11 phones, such "cross calibration" leads to errors below 4 dB in 89 % of cases and below 2 dB in 52 % of cases. We therefore conclude that, for general usage, it is acceptable to calibrate per model, rather than per individual device. This is important because we cannot expect all individual, casual users of NoiseTube Mobile to go to the process of calibrating their own device. In other words, device-specific calibration does not scale, while model-specific calibration does[41]. We also established a method to "average" the calibration points sets determined for multiple instances of a model, resulting in a single "generic" set. Through simulation we showed than such a generic set allows to make model-specific calibration more accurate.

We found no evidence of significant shifts in amplitude responses over time, hence we did not need to recalibrate the tested device. However, because we only compared responses for one phone, after a intervening period of (just) 5 months, we cannot rule out the possibility of such shifts completely without further testing.

In our validation experiments in the lab we found that device-specific white noise calibration (in combination with our linear interpolation-based correction algorithm) results in almost negligible errors for white noise below 100 dB(A). Above that level the microphone

[41] Especially because we can distribute model-specific settings from a central point [376].

of the Nokia 5230 saturates and hence accuracy worsens. Simulations also showed that the accuracy for pure tones also significantly improves after calibration using white noise.

Of course these lab experiments and simulations are not representative of conditions in the real world, in which mobile phones are exposed to the elements and in which sounds are of a very different nature than pure tones or white noise. Hence we have set up a validation experiment in the field. One important observation in this experiment was that gusts of wind caused significant errors – due to the lack of a windscreen on the phone – while in absence of direct exposure to the wind errors were very small. This underlines the importance of attaching a quality factor to collected data based on contextual factors such as the weather. To some extent that is already possible with the automatic contextual tagging feature on the NoiseTube CM (see section 5.5.1.1); as well as through social tagging[42]. Yet this is likely insufficient since, as demonstrated in the experiment, the detrimental influence of the wind can appear and disappear in a matter of meters or minutes. However, another important finding was that, precisely because the influence of the wind (and possibly other disturbances) varies constantly, its effect disappears when measurements are averaged over longer intervals (e.g. > 15 minutes). In fact over such intervals the error became almost negligible.

Based on lab experiments and simulations, we conclude that white noise calibration significantly improves the accuracy – through correction of level-dependent systematic errors – of individual SPL measurements of white noise and pure tones. In the field experiment, we found that if sufficient amounts of data are considered, errors on average sound levels all but disappear. While circumstantial this result supports the hypothesis put forward in section 7.1, namely that the errors on individual measurements taken with mobile phones – due to imperfections of the hardware (which calibration can only partially correct for), external influences (e.g. wind) and user behaviour – can be compensated through aggregation and statistical reasoning over large amounts of data with dense spatio-temporal granularity, which participatory mobile sensing – at least potentially – allows to collect at low cost. Besides averaging measurements over time, as we have done here, the same can be over space (e.g. per area unit), as we discuss in the next section.

To our knowledge this is the first time the calibration and accuracy evaluation of mobile phones as sound level meters has been carried out so extensively and thoroughly. In [438] Rana et al. report on the testing and correction of amplitude responses of 2 devices of different models without specifying the nature of the test signal. In [292] Kanjo et al. mention that they calibrated 1 device but do not specify the procedure; the resulting accuracy was only tested in an irreproducible manner (measuring the conversation of two people in an office). In [467] Schweizer et al. mention the possibility of calibration using pink noise, but do not discuss concrete experiments or results. An exception is the work of Santini et al. [459] who investigate both amplitude and frequency responses

[42] As we saw in section 5.6, some users spontaneously tag data with information on weather conditions.

of 3 phones (of the same model) using different test signals. Yet they do not give a systematic overview of the frequency-sound level domain as we have done. While they evaluated the possibilities, no actual calibration (i.e. correction) was tried out. What is interesting however is that they conducted lab experiments in which phones and a reference SLM were exposed to pre-recorded sounds, typical of urban situations. While probably not suitable for calibration purposes (due to inevitable biases), such recordings are definitely suitable for validation purposes, because, contrary to validation in the field, such experiments are reproducible and allow to study accuracy for real-world sounds in absence of real-world disturbances such as wind. Hence, it would be a useful intermediate step between white noise or pure tone validation in the lab and outdoor validation.

7.3 Participatory noise mapping

Having thoroughly tested and analysed the performance of phones as sound level meters, the next step is to use them in a realistic participatory noise mapping campaign. In this section we discuss a two-phased experiment in which a group of volunteering citizens, trained and coached by us, used the NoiseTube system to carry out a coordinated measuring campaign. As explained in section 5.3, initiatives to set up such campaigns can come from both citizens or authorities. In this experiment we have taken the perspective of citizen-led initiatives (assisted by scientists) – typically motivated by concerns about local problems, which (in the eyes of initiatives takers) are not or inadequately assessed by officials. However, most findings are equally relevant to authority-led sensing initiatives.

7.3.1 Context

We were fortunate in that we could collaborate with one of the best-known citizen-led environmental activism groups in Belgium[43], *Ademloos*[44] [5]. This non-profit association, founded in response to controversial plans for reorganising road traffic around Antwerp, has been fighting for the past ten years for a more sustainable solution to the city's congestion problems and related environmental issues including noise and air pollution. We were first contacted by Ademloos in March 2010. They had heard – through the media and word-of-mouth – about the NoiseTube project and were keen to try out the technology to collect evidence on noise pollution. We convinced them to set up a collaboration in which they would help us to validate and improve the technology, such that, in a later stage, they would be able to use it for their own purposes with our support.

[43] Or at least in Flanders.

[44] "Ademloos" is Dutch for "breathless".

In total 13 Ademloos members voluntarily participated in our experiment. We got to know these people as being highly motivated, environmentally-concerned and community-driven individuals. In other words, they were an ideal public for a community/citizen science experiment. However, most of them had little scientific domain knowledge and many had a limited level of literacy in ICT – in fact, some did not even own a mobile phone.

7.3.2 Planning & protocol

A first meeting with the chairman/spokesman and the secretary of Ademloos resulted in a list of requirements both for the citizens as for us researchers. The group had the main say in practical issues such as the measurement dates and times and the area of focus, namely the *Linkeroever*[45] neighbourhood, highlighted on the map in figure 7.1.

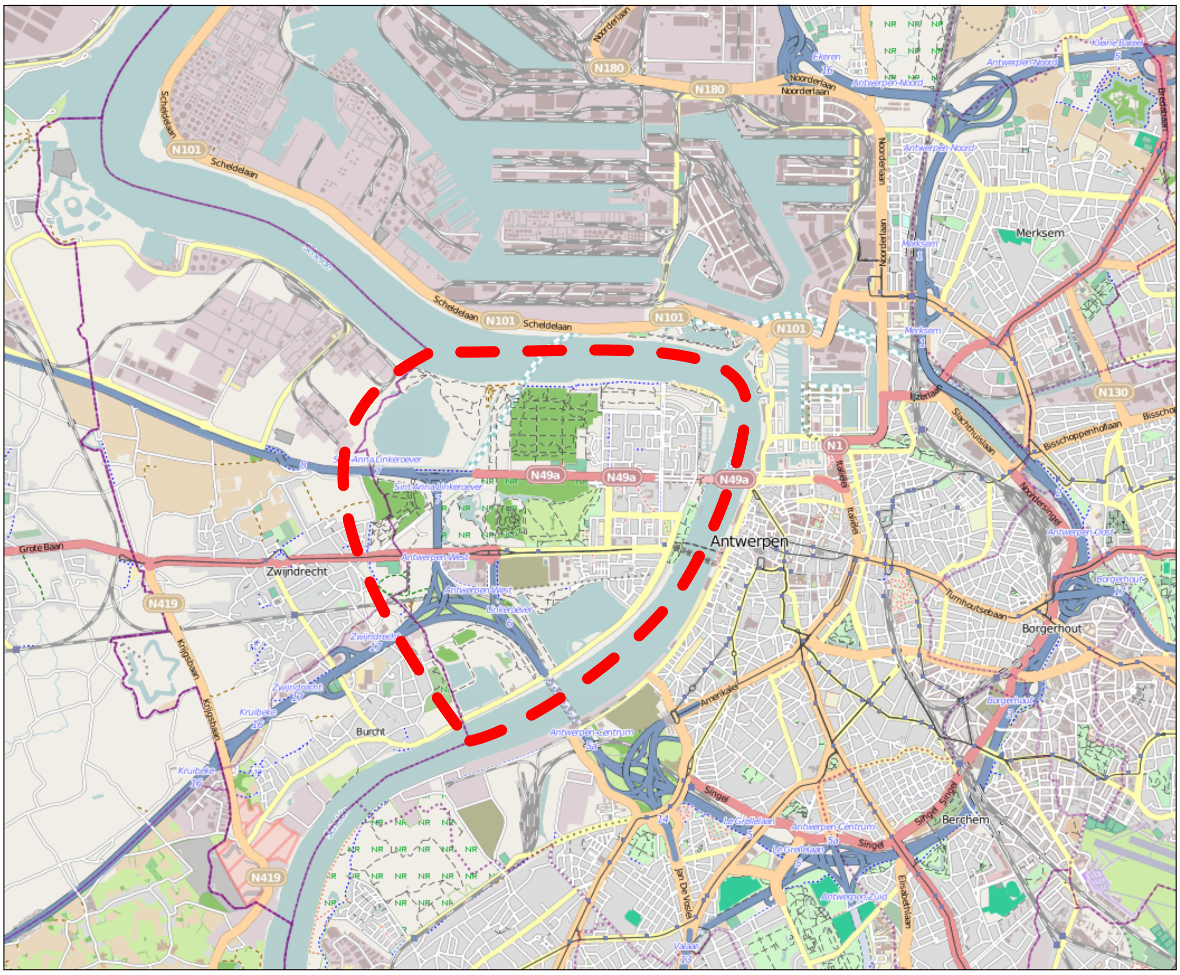


Figure 7.1: The city of Antwerp and the Linkeroever neighbourhood (circled in red)

From our side we structured the actual experiment in a way that allowed us to maximise research outcomes. As this was the first time NoiseTube was used in a coordinated

[45] Linkeroever is Dutch for "left bank".

measurement campaign, we decided to run the experiment in two phases. Phase 1 consisted of a small (in terms of participants involved and covered area), strictly coordinated experiment allowing us to test and fine-tune procedures. This phase ran over a period of 10 days. Phase 2 involved more volunteers and less constraints, thus corresponding more closely to a (loosely coordinated) participatory noise mapping effort at a city level, where data accumulates as citizens use NoiseTube where- and whenever they see fit. This phase ran for a period of 5 days. While the first phase guarantees enough data (per space/time unit) for statistical averaging to make sense, in the second phase it is more difficult to produce complete noise maps as there may be gaps in time as well as in space. However, we shall see below that the maps produced are still valuable, as they offer insight in noise pollution patterns as well in user behaviour under less constraints. Specifics and outcomes of both phases are discussed below, in respectively section 7.3.4 and section 7.3.5. In each case we make a qualitative comparison of the resulting maps with official noise maps of the area, demonstrating if and how the two can and cannot be compared, and indicating where the most obvious commonalities or differences are.

As per our suggestion, the secretary of Ademloos, who lives in Linkeroever, acted as the main communication channel between us and participants, his first task being to line up a list of interested Ademloos members located in that area. Meetings were set up at each stage of the experiment. First, we organised an introductory meeting with interested members, after which volunteers for the actual experiment phases were recruited and their main constraints summarised, which allowed us to concretise times and areas. Second, training meetings were set up right before each phase, to distribute phones and to specify how, where and when measurements were to be carried out. We gave precise instructions on how to use the phones and the NoiseTube Mobile acrapp – which was crucial as about half of the volunteers had limited experience with mobile phones, and almost none were familiar with mobile apps. In these meetings we also asked participants to adhere to guidelines related to data quality. Concretely we asked them to:

- always keep the phone in their hand with the microphone pointing away from themselves, making sure that it was never covered by clothing;
- not to use other phone functionality while measuring (e.g. type or read SMS messages, make phone calls, etc.);
- not to talk while measuring – in case they ran into other participants or acquaintances we asked them to only politely wave.

Prior to each phase users were also given a short document in which the measurement protocol and all instructions[46] were summarised in simple terms and with lots of pictures. The document also contained contact details of us, the secretary and all participants.

[46] Including troubleshooting steps, such as what to do when the application would crash.

Due to their limited experience with mobile phones we let participants work with a simplified version of NoiseTube Mobile (Java ME variant), in which the UI is entirely focused on sound level measuring. Support for social tagging was disabled because we feared it might confuse these users[47] and too much fiddling with the phones could lead to distorted measurements. Other features we removed are the ability to disable geo-tagging through GPS (as this is essential in order to make noise maps) and the ability to disable data saving (for obvious reasons). This version also included a workaround for the memory leak bug discussed in section 6.5.5.1. To keep the acrapp from crashing we let it automatically restart at regular intervals. However, because automatic restarts are mutually exclusive with Internet access, data could only be stored locally and not be sent to the NoiseTube CM in real-time. As not all participants had a home computer, or the necessary skills, we did not ask them to upload track files to the CM – say, every evening. Instead, after the experiment phase was over we collected the phones and recovered the track files from them. Because it was not possible for the participants to see the data collected by their peers during each phase and results would be discussed with the group in "offline", face-to-face meetings, we did not ask them to use the NoiseTube CM.

At a meeting in May 2011 we presented the noise maps and other outcomes resulting from both phases of the experiment, leaving ample time for discussion, feedback and questions. During this meeting we also let the attending participants fill out a questionnaire on their experience with NoiseTube and participatory noise mapping. The results of this survey are discussed in section 7.3.7.

7.3.3 Grid-based aggregation

The NoiseTube CM software currently supports the generation of 3 types of "city-level" noise maps (see section 5.5.1.3). However, none of these is really suitable to visualise the results of our Linkeroever experiments[48]. In the default type individual measurements are represented by coloured circles without any aggregation[49]. Considering the amount of data collected in this relatively small area, such a map would be overly dense and almost unreadable. But more importantly it would not be possible to visualise the *average* sound level measured at different places within the area, which is more representative and, as discussed in section 7.2.6, potentially more precise than showing individual measurements. In the other types of maps the CM generates[50] average sound levels are shown for either districts or street segments. The district map is not suitable because it would be too general – as the whole focus area falls within one district (i.e. Linkeroever). While better

[47] Due to the somewhat unintuitive way it was implemented in the Java ME variant (see section 6.3.3).

[48] Nor those of similar highly localised noise mapping campaigns.

[49] As in the maps the CM generates for individual tracks (see section 5.5.1.2).

[50] Provided that we would have added a digital map of districts and streets in Antwerp – e.g. sourced from OpenStreetMap [399] – to the database.

than the default map, the street map is not suitable either because we want to measure and show noise levels in places that are not directly adjacent to streets (e.g. in a park).

Instead we have chosen to create noise maps using an approach we call "grid-based aggregation". This means the area of study is divided by a grid with cells of equal size (e.g. 20 m × 20 m). Then each individual measurement is assigned to a cell based on its geographical coordinates. Statistics such as average, standard deviation, minima and maxima are then computed for the data in each cell. Finally a map is generated representing the average sound level measured in each cell, along with additional metadata.

One of the factors we have to take into account when deciding on a suitable size for the grid cells is the inherent error on geographical coordinates obtained from GPS. Such errors tend to vary with the time of day (due to GPS satellite positions), atmospheric conditions, and especially the density and height of nearby buildings. To get an idea of the precision of GPS coordinates, obtained on our Nokia 5230s in the relevant area, we conducted some in situ tests. We positioned ourselves at 7 locations in Linkeroever and took 100 GPS readings at each (with the phone lying on the ground). Then we computed the spread (w.r.t. the averages) on longitude and latitude coordinates obtained. We found an average spread of 2.76 m on latitude and 2.23 m on longitude. Maximal differences measured were 8.25 m for latitude and 7.94 m for longitude. We should note that this only tells us something about the precision of the coordinates, not their (absolute) accuracy[51], but this is enough for our purposes. Hence, cells smaller than 10 m × 10 m make no sense.

To aggregate the collected data (stored in track files as described in appendix D) on a grid, compute statistics and generate maps we have developed a tool chain consisting of:

- a program, written in Scheme, that parses track files coming from multiple phones and categorises all measurements (or subsets based on time periods) across grid cells, calculates statistics for each cell, and saves the result as a CSV[52] file;
- a program written in Java which converts such CSV files to maps in either the *Shapefile* [152] format – for use in desktop GIS software – or the KML [298] format – for use in Google Maps/Earth [215, 217] or other online GIS applications;
- a Web application written in Javascript and PHP, using the OpenLayers library [398], that allows the KML maps to be consulted online in an interactive manner – see [65]. Users can click on grid cells to see further statistics such as minimum, maximum and average sound levels, standard deviation, and sample size.

For practical reasons this tool chain was developed separately from the NoiseTube CM. However, we see it as a prototype for features that will likely be integrated in the CM software later on.

[51] To assess accuracy we would need to compare with reference points on the ground or a trusted reference device such as a Differential GPS [580] receiver.

[52] *Comma-Separated Values*.

7.3.4 Phase 1

In the first phase of the experiment we wanted to control as many free parameters as possible in order to be able to focus on the quality of the collected data. Therefore we have chosen to let our volunteers measure noise along a fixed, predefined route, at fixed, predefined times. While this is but one, rather dull way to coordinate measurement campaigns, we felt it was the best way to guarantee enough data would be collected for a limited number of specific times and places.

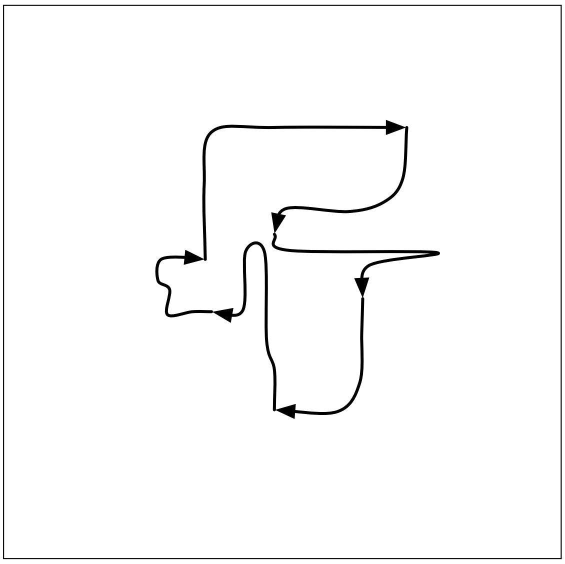


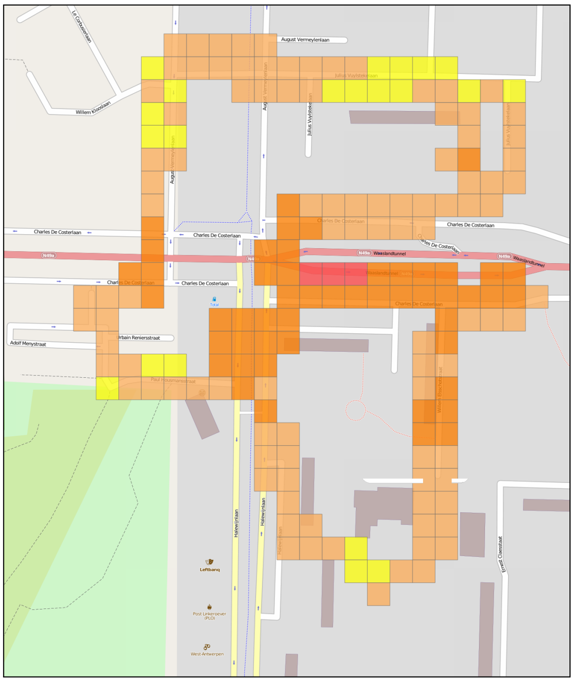
Figure 7.2: Walking route for phase 1

The chosen route, which is shown on the map in figure 7.2, is circular, covers a distance of about 2 km and is confined within an area of roughly 400 m × 400 m centred around one of the busiest intersections in Linkeroever. It was mapped out to contain a composite of typical urban and suburban soundscapes, such as a busy intersection, a park, and residential streets – all near the area were the volunteers lived. Walking the route takes about 30 minutes. Its length was chosen such that that four people, measuring one hour a day (walking the route twice) would produce enough data so as to represent the statistical sample space adequately. We did not define a fixed starting point along the route. Hence each participant could pick his/her own. This way we avoided having too much overlap in the measurements – which would be the result of 4 people walking the same route within a few meters of another – and we limited the temptation for participants to talk while measuring. Moreover it allowed participants to limit the distance they needed to walk by starting from a point close to their home.

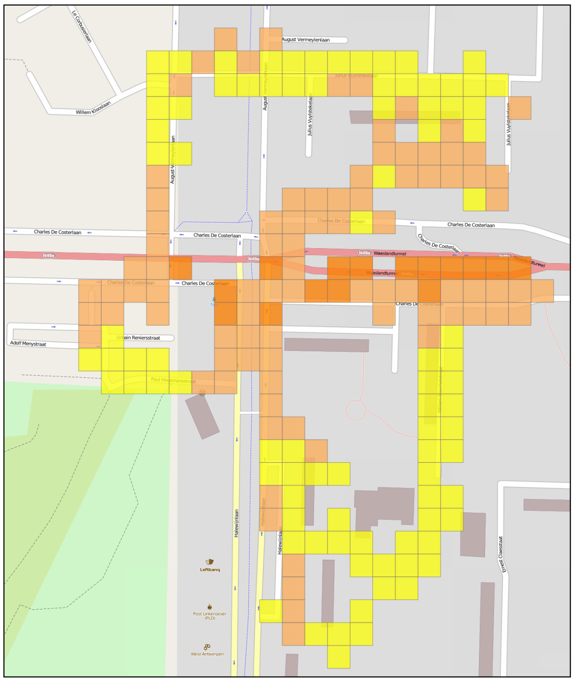
The experiment spanned 2 working weeks in July 2010. During the first, four people were asked to walk the route twice between 21:00 and 22:00, outside the peak hours for road traffic, for 5 consecutive days. The next week four different people did the same between 7:30 and 8:30, a peak hour for road traffic. A quick calculation shows that in theory a total of about 36000 measurements should be gathered in each week[53], which gives an average of 180 measurements per 10 m of the along the route – ample for statistical analysis.

In figure 7.3 we see two noise maps, generated with the tool chain discussed above, based on data collected during phase 1. The map in figure 7.3a shows average sound levels for the peak hour and is based on 30977 measurements, while the one in figure 7.3b shows

[53] 30 measurements / minute × 60 minutes × 5 days × 4 people.



(a) Peak hour
(7:30–8:30; 20 m × 20 m; min. 50/cell)



(b) Off-peak hour
(21:00–22:00; 20 m × 20 m; min. 50/cell)

(c) Official L_{den} map (detail)

Figure 7.3: Phase 1 maps, (a) and (b), using the colour scale of the official L_{den} map for road traffic, the corresponding detail of which is shown in (c)

average sound levels for the off-peak hour and is based on 36394 measurements. Hence for each week the number of measurements is close to the theoretical estimate above. Measurements were made during different days of the week, but are aggregated together to obtain averages for the chosen hours. On each map the chosen route is clearly recognisable. Both maps use the same colour scale as the official L_{den} map for road traffic in Antwerp [557], the corresponding detail of which is shown in figure 7.3c.

For these maps we have chosen a grid with cells of 20 m × 20 m, excluding cells with less than 50 measurements to ensure significance. Note that there is always a trade-off between these values, as a smaller grid size requires more data to achieve statistically significant results. Moreover, it does not make sense to use grid cells smaller than the error margins of the GPS positioning. The maximal GPS errors mentioned above suggest that one could increase the resolution of the grid up to cells of 10 m × 10 m. However, by experimenting with values for cell and minimal sample size we found the above combination to be best in terms of balancing clarity with significance. Table 7.3 summarises statics about both maps. The difference of 2.8 dB in average sound level is reflected by a clear visual discrepancy between the maps for both time periods – e.g. there is much more yellow on the off-peak map. High noise levels along the main busy road – traversing west–east through the middle of the area – are clearly recognisable on each map. An interactive version of these maps, and others generated from data collected in phase 1 (e.g. with different grid cell sizes), can be consulted online [65].

	Peak hour map	Off-peak hour map
Number of grid cells (meeting the requirement of ≥ 50 measurements)	172	192
Average # measurements / cell	164	167
Minimum average sound level	56.7 dB(A)	55.3 dB(A)
Maximum average sound level	71.9 dB(A)	69.7 dB(A)
Average average sound level	63.6 dB(A)	60.8 dB(A)
Average standard deviation	5.2 dB(A)	5.0 dB(A)

Table 7.3: Statistics about the peak and off-peak hour noise maps for phase 1

Comparison with official noise maps for the area is difficult due to the difference in approach (measuring vs. modelling) and in the quantities that are represented. As per the END [173] requirements[54] the official strategic noise maps for the city of Antwerp [557] represent either L_{den} or L_{night}, and this for road traffic, air traffic, train traffic or industry separately. Hence there are 8 different maps, all generated using the simulation method discussed in section 4.3.2.8. Ideally we should compare our peak and off-peak hour maps with respectively an L_{day} and an L_{evening} map. Unfortunately such maps are not mandated by the END and have therefore not been made – or at least not published – for the city of Antwerp. Nevertheless it is useful to attempt at least a qualitative comparison, if only to highlight immediately obvious commonalities and/or differences. The most suitable map to compare with is the L_{den} map for road traffic, shown in figure 7.3c. This is because first, the main noise source in this area is without any doubt road traffic, there being no airport or railways in the immediate neighbourhood and industry also being somewhat farther away. Second, because all our data was collected during the day and the evening comparison with the L_{night} map would be flawed. However, because L_{den} is a weighted average[55] it is one step further away from (measured or simulated) L_{Aeq} values. The map

[54] See section 4.3.2.7.

[55] Of L_{day}, L_{evening} *and* L_{night}, with a bias for evening and night noise (see section 4.3.2.2).

dates from January 2010 but relies on traffic statistics going back to 2006. It represents the L_{den} noise level in an average 24 hour day, at a height of 4 m above the ground – again per END requirements. Whereas our maps are based on measurements of *all* noise (or rather sound) sources, taken at a height of about 1 to 1.5 m.

Despite the differences in approach and represented quantities[56] there are a few noteworthy observations we can make by visual comparison. First and foremost we notice that in terms of overall sound level distribution our maps are quite similar to the official map. While this is not a proof of accuracy (see below), it is certainly reassuring, especially because the general patterns align well with expectations – e.g. the fact that the main west–east road and the intersection are the noisiest places. While there is little doubt that simulated maps can capture such general trends well, this has not been demonstrated before for participatory noise maps. Yet apart from the overall similarity there also some marked differences between our participatory noise maps and the official, simulated one.

One particularly interesting difference appears in the park area in the south-east quadrant, for which our maps indicate significantly higher noise levels than the official map. We think this cannot be explained simply by the difference in represented quantities. Instead we see 3 possible explanations: (1) an overestimation, by our maps, of the actual average sound level in this place, or at least the portion of it that is caused by road traffic; (2) an underestimation, by the simulated map, of the road traffic noise that reaches this area from surround streets; or (3) a combination of both (1) and (2). Explanation (1) could be due to the lower height (i.e. closer to the traffic), the influence of the wind, the presence of other sound sources, or the behaviour of the measurers themselves. Yet the influence of the wind is unlikely to have been constant during both weeks, and should thus be largely averaged out given the amount of data collected. The same goes for talking and other occasional disturbances caused by the measurer. We can also more or less rule out the presence of a constant, secondary sound source (besides road traffic) related to the location itself. What is more plausible is that measurements taken in this place were to some extent influenced by the measurers' footsteps. While the sound level caused by footsteps varies – e.g. depending on the walking pace/technique of the person, his/her footwear, the nature of the surface or objects on it[57] – the sound never entirely disappears as long as one keeps walking. Hence, there where the ambient sound level is fairly quiet[58], as one would expect in this particular place, footsteps may cause slight, yet systematic[59] overestimations[60]. However, comparison of our two maps reveals that the average sound level measured in the park was significantly higher during the peak hour

[56] I.e. peak/off-peak averages vs. L_{den}.

[57] For instance, one participant noted that the crisping sound of dry leaves under his feet caused measurements to peak.

[58] In noisier spots – e.g. next to a busy road – the sound of footsteps would be drowned out.

[59] I.e. which cannot be averaged out.

[60] We should note that the effect of footsteps was not investigated in the validation experiment discussed above, as both the phone and the SLM were exposed to their sound.

than in the off-peak hour. Because footsteps sound equally loud in the morning as they do in the evening there must be another factor, one that is, given the timeframes, is in all likelihood related to a difference in traffic intensity on nearby roads – which points in the direction of explanation (2). Hence, apart from the difference in height, we find little support for explanation (1). While we do not have sufficient data to be certain, we are inclined to believe that explanations (2) or (3) are the more likely hypothesises.

Another noticeable difference is that the average sound level our volunteers measured along the main busy road is actually somewhat lower that what is shown on the official, simulated map (especially in the middle of the road). This is however easily explained. While our participants only walked on the sidewalk next to this particularly wide road, the simulated map also shows the sound level on (or rather 4 m above) the road itself. Moreover while our maps use grid cells of 20 m × 20 m, the simulation model internally uses a grid of 10 m × 10 m. This goes to show that there will always be places (as well as times) for which participatory sensing cannot provide data, whereas simulation models can – which underscores the complementarity of both approaches.

7.3.5 Phase 2

For the second phase of our experiment we imposed a less strict coordination. Instead of specifying a fixed route and timeframe we allowed participants to measure at will – albeit within a particular area and with a daily minimum, so as to ensure enough data was gathered. Loosening up space-time restrictions is reasonable because the results obtained during phase 1 indicate that the data gathered with NoiseTube Mobile is credible. Instead of data quality we can now focus on data collection patterns, for instance to get an idea of how the freer movement of contributors affects the completeness of the maps. In comparison with phase 1 this situation corresponds to a more realistic NoiseTube use case, namely a participatory noise mapping campaign at a city level, loosely coordinated by a local NGO or authority. In such campaigns data accumulates as a larger group of citizens (with varying levels of dedication and willingness to coordinate their efforts) use NoiseTube in the context of their daily lives. Hence the data gathered in phase 2 is likely also more representative of what Linkeroever residents experience on a daily basis.

Figure 7.4: Focus area for phase 2 (framed in red)

Concretely, we asked 10 volunteers to measure for at least 1, not necessarily continuous, hour a day, without mandatory time slots, during 1 working week in November 2010 and in an area of about 1 km × 1 km – encompassing that of phase 1. All participants lived within this area, which is shown in figure 7.4. This larger effort should in theory result in at least 90000 measurements[61], which gives an average of just 36 measurements per grid cell over the whole week if we use a grid of 20 m × 20 m as above. Of course this number is just an indication since measurement activity is unlikely to be uniformly distributed over the area; still, it seems likely we may have to increase cell size to compensate for sample size.

When the experiment was over a total of 84309 measurements had been gathered, which is fairly close to what we predicted. As illustrated by chart 7.8, almost all the data was collected during the day and the evening, with peaks at 11:00, 17:00 and 22:00, and dips around 13:00 (lunchtime) and 19–20:00 (dinnertime). Unsurprisingly, very little measurements were made between midnight and 8:00.

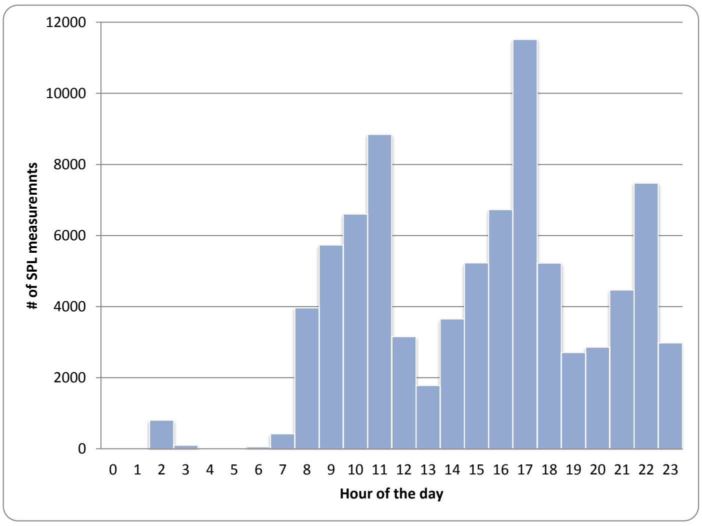


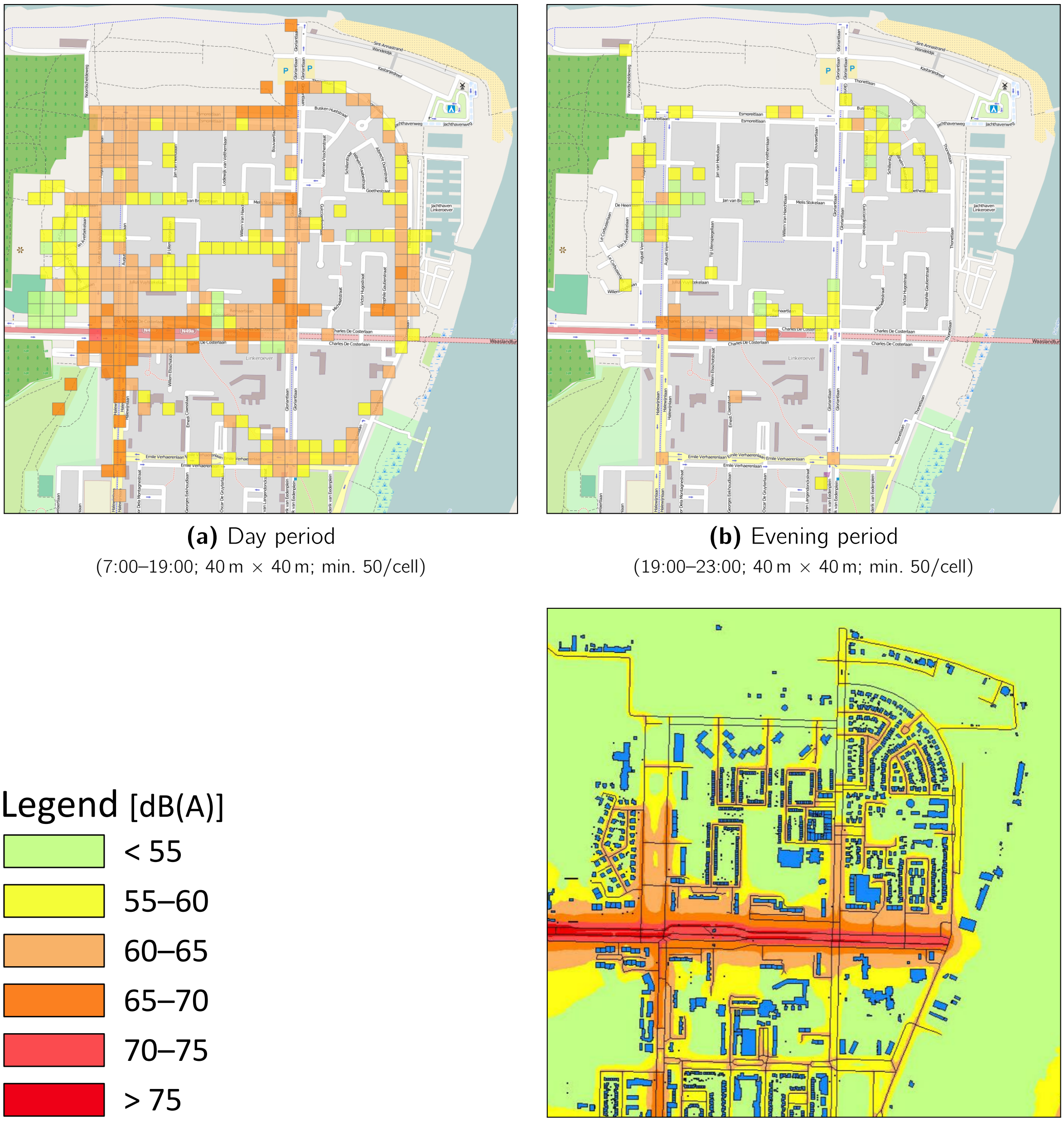
Chart 7.8: Data collected per hour of the day in phase 2

As the total data set is too small to study shorter intervals, while still maintaining significance, we focus on the day and evening periods. Consistent with the way the END is implemented in Flanders [353, 556: p. 10], we let the day period run from 7:00 to 19:00 and the evening period from 19:00 to 23:00. Concretely, 62853 measurements were made during the day, and 17513 during the evening. With these numbers and the larger area to cover we found the best map representation to be one with grid cells of 40 m × 40 m and a minimum of 50 measurements per cell. Even so, the resulting map for the evening period, shown in figure 7.5b, is very sparse. While many gaps remain, the map for the day period, shown in figure 7.5a, covers a much greater portion of the focus area. Statistics about both maps are summarised in table 7.4. Our maps use the same colour scale as the official L_{den} map for road traffic in Antwerp [557], the corresponding detail of which is shown in figure 7.3c. Again an inter-

	Day period map	Evening period map
Number of grid cells (meeting the requirement of ≥ 50 measurements)	329	90
Average # measurements / cell	77	76
Minimum average sound level	44.0 dB(A)	38.2 dB(A)
Maximum average sound level	70.5 dB(A)	68.8 dB(A)
Average average sound level	61.1 dB(A)	58.0 dB(A)
Average standard deviation	4.5 dB(A)	5.4 dB(A)

Table 7.4: Statistics about the day and evening period noise maps for phase 2

[61] 30 measurements / minute × 60 minutes × 5 days × 10 people.



(a) Day period
(7:00–19:00; 40 m × 40 m; min. 50/cell)

(b) Evening period
(19:00–23:00; 40 m × 40 m; min. 50/cell)

(c) Official L_{den} map (detail)

Figure 7.5: Phase 2 maps, (a) and (b), using the colour scale of the official L_{den} map for road traffic, the corresponding detail of which is shown in (c)

active version of our maps can be found online, complemented by a number of alternative maps, based on the same data but focusing on different users and time intervals [65].

For comparison we again juxtapose our noise maps with the official L_{den} map for road traffic. The choice remains unchanged because almost no data was collected at night and – also in this larger area – road traffic is the main source of noise. For the same reasons as discussed above comparison of our maps with official ones is difficult. However it further complicated due the fact that the data for phase 2 is much sparser, especially in the evening. Therefore we only attempt a visual comparison of our day period map.

Again we see areas for which our map shows markedly higher noise levels than the official map does. The clearest example is the northernmost west-east road on the map. According to the official map road traffic noise along this road (which was likely not modelled) stays below 55 dB(A), but the data collected by our measurers strongly refutes that. The same holds, to a lesser extent, for the areas in the north–east and south–east corners of the map. It is precisely in such areas, away from the busiest roads (which the simulations capture well), where participatory sensing is able to make a difference.

Another thing we notice is a zone, towards the west side of the map and just north of the main west–east road, for which our day period map shows sound levels that are suspiciously (given the road's proximity) lower than on the official map. Upon inspection of the raw data we learned that almost all measurements assigned to these cells originate from a single, very long track. Visualising the track revealed that the measurements in question were almost certainly taken while the phone was in a quiet, indoor place[62]. Of course it is trivial to remove this sort of faulty data from the set and then generate a new map. However, we did not do that because this case is a helpful reminder of the fact that unintentional user behaviour[63] is a potential source of anomalies, especially if there is little spatio-temporal overlap in the data collected by different people and/or at different times. A general solution is to collect much more data at the same as well as different times and places, ideally by multiple people. But if scaling up the campaign is not possible, coordinators, possibly aided by technology[64], need to be wary of suspicious patterns – but one could argue that the same is true for simulated maps.

Finally we should note that the gaps in the day map, and eventually also in that for the evening, would likely have become much rarer if we had increased the number of participants and/or the duration of the experiment. However, without very strict coordination (e.g. "person A must measure in streets M and N between times X and Y"), or a truly massive number of participants, it may be difficult to fill *all* gaps – i.e. to achieve full coverage of the area. One way to fill gaps without measuring may be to apply spatial interpolation techniques. However, as noted in [159: p. 45] and confirmed by our own unsatisfying attempts, standard interpolation algorithms offered by GIS software[65] are not suitable for modelling sound propagation. A more suitable solution could be to fill gaps in participatory maps with data from simulated maps – which once again indicates that both noise mapping approaches can be complementary. For obvious reasons it may also be infeasible to create participatory maps for night noise, especially over large areas. Hence also for that purpose the simulation approach remains useful and necessary.

[62] When a GPS receiver does manage to get a signal indoors the obtained coordinates tend to be very jittery, which explains why these measurements are scattered over a relatively large area, even though the device was not moving.

[63] In this case the person probably simply forgot to stop measuring when he or she came back home.

[64] For instance it should be feasible to develop an algorithm that detects whether a device spent time indoors based on jittery GPS coordinates.

[65] E.g. *Inverse Distance Weighting* (IDW) and *Kriging*.

7.3.6 Assessing noise map accuracy

Through visual comparison of our maps for phase 1 and 2 with the official L_{den} map, we have found strong indications that support the validity (e.g. the capturing of expected trends) and value (e.g. the detection of traffic noise that is apparently underestimated by the official map) of our participatory approach and its complementarity with the conventional simulation-based method.

Visual observations alone are not enough to draw firm conclusions about the accuracy of our maps – nor of official ones. Besides the differences in represented quantities, another complicating factor is that, due to the colour bands of 5 dB, a lot of spatial variability (which may or may not reflect reality) remains hidden. Based on the data collected in phase 1 and 2 we could easily generate maps with narrower bands, or compare the aggregated data numerically instead. Yet for the official map that is not possible because the responsible authorities do not provide access to the data behind the map[66], hence we are stuck with the end product. Still, even if we had access to raw data, or even if we had official L_{evening} and L_{day} maps to compare with instead of the L_{den} map, that would not mean we could use it as a "reference"[67] – because it is based on source-specific simulation, not actual measurements.

It would be interesting to know the (estimated) error margins on the output of the simulation model, but to our knowledge no such information has been released with regards to the Antwerp noise maps. In the case of Brussels, the BIM/IBGE, the regional environmental agency, claims that the errors on their END-type noise maps are within ± 2 dB [64: p 11]. While such error margins can be estimated [159: pp. 105–109], the BIM/IBGE bases them on comparisons with actual L_{den} or L_{night} measurements made using the sensor network they operate [429] – which once again stresses the importance of having a proper reference. If we assume that the Antwerp maps have similar error margins[68], then this is not better than the error of < 1 dB we found over the duration of our validation experiment discussed in section 7.2.5. However, we should conduct more such side-by-side experiments before we can make firm claims about the error margins of our maps. One way to assess accuracy in future noise mapping experiments would be to collect reference measurements for a subset of the data – by letting some of the volunteers use NoiseTube Mobile side-by-side with a SLM, or by temporarily placing stationary SLMs at spots where measurers pass by. Another way to assess uncertainty (also for the data we already have) may be to apply descriptive statistics techniques, such as cross-validation and split-sample validation, which we have yet to look into.

[66] In fact our contacts in the Antwerp city administration tell us that even they cannot easily access this data because the actual noise maps are produced by a subcontractor.

[67] As we used a calibrated SLM as a reference to assess the accuracy of measurements made using NoiseTube Mobile on a single phone.

[68] Which is not necessarily the case given the fact that the Antwerp maps are the responsibility of another agency and, at least in Linkeroever, the urban density is much lower than in Brussels.

7.3.7 User study

We consider it important to also evaluate NoiseTube, as a system for participative citizen-led noise mapping campaigns, from the user perspective. Therefore we need an evaluation tool. The feedback gathered with this tool can be used to guide further development of NoiseTube and similar participatory sensing systems for environmental monitoring.

Considering the limited number of people who took part in our Antwerp experiments we could have taken an interpretative qualitative approach to evaluate the experience during the different phases. But since we plan to reuse the evaluation tool with a larger audience (e.g. students who tested NoiseTube or registered users of the NoiseTube CM) we have chosen a methodology that can be scaled up more easily, namely that of a standardised questionnaire. Hence, the Ademloos members are not only a pilot group for testing the NoiseTube system but also for the evaluation tool. To design this questionnaire we built on dimensions found in literature on participative/mobile sensing, in combination with open issues regarding data representation [22, 98]. We kept some open questions since we are still exploring the dimensions of the experience of sensing. Building on the results with the Ademloos group, and possibly other pilot groups, we should be able to revise the questionnaire to have closed categories for these dimensions as well. The questionnaire (which is in Dutch) is included in appendix F.

The questionnaire was filled out by the group at the start of the final feedback session in May 2011, during which we presented the results of the measuring campaigns. All 13 participants of the experiment completed the questionnaire – those who could not attend the meeting filled it out at home. These are 7 men and 6 women with an average age of 62 (standard deviation of 13 years). From a methodological point of view we learned that some questions should be reconsidered because the answers do not show a lot of variation. For example a ranking question, instead of a Likert item scale, would have given more insight in prioritisation of the kind of information to be shown on noise maps, as well as on the motivation to take part in a mapping campaign. Although this was a first test of the questionnaire as an evaluation tool, besides methodological insights it has also provided valuable feedback concerning the NoiseTube system as it was tested. Below we summarise the most important indications. We must however stress that statistical significance is not possible with this limited and opportunistic sample of testers. Hence, while the feedback provided by this particular group is certainly helpful, it cannot be used to draw general conclusions.

Looking at the motivational factors for taking part in the campaign, most agreement was on a general concern about noise pollution (11 out of 13 considered it "very important"), followed by personal experience of noise pollution ($\frac{8}{13}$ indicated this was "very important"), and supporting the Ademloos activism group (also $\frac{8}{13}$ "very important"). Supporting scientific research was also seen an important ($\frac{5}{13}$) or very important (also $\frac{5}{13}$) motivation.

Factors which were less important drivers for this particular group were an interest in technology, or that the campaign was a fun or a useful pastime. When asked about possible concerns regarding privacy, only one participant said he was worried because « *there is a log of where I was at which moment* ». A lack of general concern about privacy can be hypothetically attributed to the fact that participants were part of an existing group that know and trust each other[69], who consciously took part in a scientific experiment and who got time to get acquainted with the researchers is person.

With open questions we asked for the 3 most pleasant and 3 most annoying things they experienced taking part in the campaign. Some unexpected dimensions appeared. The immediate feedback provided by the acrapp was seen as pleasant because it gave insight in sound (level) and in the problem of noise (relativity, locality). Several people mentioned that they liked the team spirit the campaign created in their group. Multiple participants enjoyed the fact that walking was an integral part of the sensing activity, which was seen as beneficial for physical fitness and general health. However one participant found it annoying that the campaign obliged him to walk through unhealthy areas polluted by heavy traffic. Less annoying experiences were mentioned than pleasant ones. Other annoying experiences were: stability of the app[70], the need to constantly hold the phone, the dullness of fixed routes (in phase 1), and not being able to have conversations while measuring (one person even noted that she regretted having to pass by acquaintances, which indicates how committed she was to the experiment). Regarding the fixed hours in phase 1 and the freely chosen times in phase 2, opinions seemed to be mixed. For instance, one user who took part in both phases mentioned that the fixed timeframe was easier to keep up (comparing phase 2 to 1), while another who also took part in both phases mentioned the fixed timeframe as an annoying aspect (of phase 1).

We also asked for feedback on features of the NoiseTube system which these users had not been given access to. For instance, we left the social tagging feature out of the acrapp used by the participants. In the questionnaire we asked whether they would have liked to be able to indicate sources of noise while measuring. The group was very positive about this ($\frac{10}{13}$ answered "Yes"). Also popular was the idea of being able to comment on the measurements made by their group through a website ($\frac{6}{13}$ "Yes", $\frac{3}{13}$ "Maybe"). The proposed possibility of commenting on measurements made by other NoiseTube users (outside of their group), was seen as much less interesting ($\frac{8}{13}$ "No", $\frac{1}{13}$ "Maybe").

Finally we asked questions about the information that is, or could be, presented on noise maps. We found two interesting things. On the one hand, sound level was preferred ($\frac{9}{13}$) to be displayed in categories (as on conventional noise maps) rather than as exact values. On the other, a slight majority ($\frac{7}{13}$) preferred maps to show peak values rather than averages, which may indicate a flawed understanding of the dynamics of sound perception.

[69] In line with common expectations for mobile sensing at a group scale, as discussed in section 3.3.3.

[70] Related, for the most part, to due to the automatic restarts which were confusing and sometimes failed.

7.4 Conclusion

The challenge of collecting data of acceptable quality, primarily in terms of accuracy and precision, represents the main hurdle that must be overcome for participatory sensing to be accepted as a suitable method for environmental monitoring. We hypothesised that this can be achieved by rigorous calibration of devices (to reduce systematic errors) and by statistical reasoning over large amounts of data with dense spatio-temporal granularity (to reduce random errors and increase representativeness). To test this hypothesis, and thereby validate our participatory approach for the assessment of environmental noise, we have conducted a series of laboratory and real-world experiments.

Mobile phones as sound level meters

By means of pure tone and white noise experiments in the lab and subsequent simulations we confirmed the two main assumptions that underpin the way measurement correction is implemented in NoiseTube Mobile. First, we found that the frequency responses of the tested phones are sufficiently flat for level-dependent corrections to give reasonably accurate results. Hence the decision not to develop digital filters is appropriate. Second, we found that amplitude responses of different instances of a single mobile phone model are very similar and that, for general usage, it is acceptable to calibrate per model, rather than per individual device. We also established a method to "average" the calibration points set determined for multiple instances of a model, resulting in a "generic" set that allows to make model-specific calibration even more accurate. Based on validation experiments in the lab and additional simulations we concluded that white noise calibration significantly improves the accuracy of SPL measurements made by NoiseTube Mobile, although we do not meet the formal requirements for Class 1 or Class 2 SLMs. However, the field validation experiment, conducted in a realistic, outdoor setting, demonstrated that when measurements are averaged over longer intervals errors with respect to a Class 2 SLM are negligible. While still circumstantial this result clearly supports our hypothesis.

These reassuring findings, as well as the fact that the calibration and validation of mobile phones as sound level meters has never been carried out so extensively before, makes this work one of the main contributions put forth in this dissertation.

Participatory noise mapping

To evaluate our participatory noise mapping approach in practice we set up a two-phased experimental campaign which ran for 3 weeks in the city of Antwerp. We collaborated with a local environmental activism organisation, 13 members of which volunteered to carry out measurements using NoiseTube (on calibrated phones) in an area of $\pm 1\,\text{km}^2$.

In the first phase we focused on the quality of the collected data. To ensure dense spatio-temporal granularity, we let our volunteers measure noise along a fixed, predefined route, at fixed, predefined times. In the second phase we focused on data collection patterns in a more realistic, loosely coordinated campaign involving more volunteers and less constraints. The data collected in both phases was used to produce noise maps for different timeframes using a grid-based aggregation technique. By averaging measured sound levels per grid cell and discarding cells that contain insufficient numbers of measurements, we obtain values that are more representative of the typical sound level at each place/-time than individual measurements are, while at the same time controlling for random errors (e.g. due to the wind or measurer behaviour) and thus improving data quality.

While additional analysis and experiments are required before we can make quantitative statements about the absolute accuracy of the resulting maps, qualitative comparison with the official road traffic noise map of the area provided strong indications that support the validity (e.g. the capturing of expected trends) and value (e.g. the detection of traffic noise that is underestimated by the official map) of our participatory approach, as well as its complementarity with the conventional simulation-based method. This result demonstrates that, when taken seriously (i.e. using calibrated phones, ensuring spatio-temporal density, etc.), participatory sensing can serve as an alternative or complementary approach to official END-mandated strategic noise mapping efforts – especially when the focus is on specific local issues occurring in reasonably-sized areas or timeframes. While there are several other (research) projects which, like NoiseTube, focus on applying participatory sensing to the assessment of environmental noise (see section 6.6), to the best of our knowledge none of our peers have conducted experiments to validate participatory noise mapping at this level of scale and realism before we did. Therefore, this work represents another of the main contributions of this dissertation.

In order for participatory environmental monitoring to be successful it is crucial that we take end-user opinion[71] into account in the ongoing effort to improve the NoiseTube approach and system. As a first step we have carried out a small user study in which we polled volunteers for their motivations to participate, their experiences in using the tool, and their opinions about noise assessment and mapping in general. Even if this survey is too limited to draw general conclusions, it does demonstrate the motivation and ability of untrained citizens to participate in this type of community science campaigns.

The noise mapping experiments and the user study undertaken in Antwerp represent research that is radically different and an essential complement to the analysis – discussed in sections 5.6 and 5.7 – of the largely uncoordinated usage of the NoiseTube system by hundreds of anonymous individuals across the world. For one thing, here we have worked directly with a group of likeminded neighbours who have a real stake in the shared commons that is the soundscape of their neighbourhood. For another, the campaign was

[71] Regarding the usability of the tool but also the conceptual and organisational aspects of the campaigns.

locally coordinated, primarily through face-to-face meetings. This has allowed us to study a situation that is much closer to the initial community memory vision (see section 2.5) and the group sensing scale (see section 3.3.3). Moreover, by taking control of more parameters – e.g. choice and calibration of devices and measurement protocol – we have followed a more direct route to reproducible and comparable results. Today these results are already helping us to convince other citizens' organisations and authorities of the potential of participatory noise mapping (see chapter 8). Last but not least, the first-hand experience obtained in setting up the campaign and the feedback provided by participants (informally or through the questionnaire) enable us to make informed choices about future improvements, extensions or adaptations of the NoiseTube system, and about organisational aspects of future experimental or operational noise mapping campaigns.

Acknowledgements

For the laboratory experiments discussed in section 7.2 we were fortunate to collaborate with Prof. dr. ir. Patrick Guillaume of the VUB's Department of Mechanical Engineering. We would like to thank him for his advice and for letting us use the acoustical lab facilities of his department.

The participatory noise mapping experiments discussed in section 7.3 would not have been possible without the enthusiastic efforts of the members of Ademloos. First and foremost we would like to thank the 13 residents of Linkeroever who voluntarily and faithfully carried out the fieldwork. Among them was Guido Verbeke, secretary of Ademloos, whom we also wish to thank for recruiting and motivating the other volunteers and for acting as an intermediary between us and them. We also want to thank Wim Van Hees, chairman/spokesman of Ademloos, for initiating and supporting this collaboration.

Finally we would like to thank Philippe Verstichel for providing helpful comments on the paper [127] upon which this chapter is based.

Chapter 8

Conclusion

In this concluding chapter, we revisit our research goals and we highlight the main contributions of our work, and reflect on its scientific and societal impact. We also provide an overview of on-going and future research efforts related to our work.

8.1 Restating the goals

Our research is driven by the ambition to make a meaningful contribution to the search for solutions to achieve sustainable development at global and local scales. More concretely, we investigate how the latest developments within computer science and ICT can be applied to establish tools and practices that allow us to better understand, manage and ultimately protect our environment, and thereby our quality of life. As explained in chapter 2, most sustainability problems can be seen as examples of the overexploitation of a *commons*. There is a growing consensus among social scientists and policymakers about the fact that sustainable exploitation of commons, and the tackling of environmental issues in general, requires broad participation and awareness of the general public.

This led to the following problem statement, originally presented in section 1.2:

How can contemporary ICT be applied to establish participatory, low-cost tools and practices that enable communities to monitor, raise awareness about, and sustainably manage the commons they rely on?

Box 8.1: Problem statement of this thesis

In section 1.3 we set out the three principal goals of our research. We repeat them here:

Goal 1: Formulate a general approach

It is our ambition to formulate a general approach that constitutes an answer to the problem statement. This approach should be abstract enough to transcend the context of specific communities facing specific commons issues, but at the same time concrete enough to serve as a blueprint for solutions that can be implemented and deployed in practice to help specific communities deal with specific problems.

Goal 2: Apply it to a specific, socially relevant case

We intend to conduct applied, interdisciplinary research aimed at turning our general approach into a concrete, ICT-based solution for a specific, socially relevant problem. To design and implement this solution we must apply state of the art technology and make advances where necessary. This effort is interdisciplinary because it requires us to gather domain knowledge relevant to the chosen case, in order to understand the needs of potential users and the expectations of domain experts. We intend to deploy and validate this solution in real-world conditions to prepare for operational – as in, non-experimental – usage in the relatively short term.

Goal 3: Aim for broad societal impact

Driven by a sense of urgency, regarding the sustainability challenges humanity faces, it is our ambition to conduct research that has real societal impact early on. This way we intend to contribute to the raising of public, academic and governmental awareness about the specific problem we focus on, as well as sustainability challenges in general. This influences the way we conduct research, as well as how, where and to whom we communicate about it.

In section 3.4 we listed nine challenges for mobile sensing research and applications. As discussed in section 3.5.2, tackling or taking into account the following subset of 5 challenges is of primary importance in order to achieve our goals:

Challenge 1: Putting mobile sensing into practice

Challenge 2: Recruiting & retaining users

Challenge 3: Collaboration & coordination

Challenge 4: Data quality

Challenge 5: Reusable components

8.2 Main contributions

Here we list the seven main contributions we have made in this dissertation, and relate them to specific goals and challenges.

The first contribution tackles **goal 1**:

Contribution I: Community memories for sustainable societies

By following guidelines distilled from theory and practice regarding the governance of commons and environmental policy in general (section 2.3), and by taking advantage of the opportunity presented by recent social, technological, scientific and cultural trends (section 2.4), we have developed a **vision** for ICT-supported solutions that enable citizen communities to lead or participate in the governance of commons they are concerned with (section 2.5). The principal elements are **community memories** – as central data repositories and points of interaction for community members and other stakeholders – and the novel combination of **mobile sensing** and **social tagging** – as a low-cost means to collect quantitative and qualitative data about the state of the commons and the health, well-being, behaviour and opinions of those that depend on it. Based on a thorough review of the state of the art and open challenges in the field of mobile sensing (sections 3.3 and 3.4), we have refined our vision into a **general approach**, involving specifications for mobile sensing systems in a community memory context, as well as a prioritisation of research challenges to tackle (section 3.5). What is required are participatory, multi-scale systems that apply mobile sensing and social tagging to monitor local environmental conditions as well as the health and well-being of citizens. Our vision and approach transcend specific commons challenges, and are thus relevant for other (research) projects concerned with mobile sensing in environmental contexts.

To tackle **goal 2**, we have focused on the problem of *environmental noise*, commonly referred to as *noise pollution*. This work has resulted in the following six contributions, each of which also relates to one or more of the primary challenges outlined in section 3.5.2:

Contribution II: The NoiseTube approach

By building on contribution I and applying the gathered domain expertise – regarding physical aspects of sound (appendix A); and cultural, psychological, health, socio-economic and policy aspects of (environmental) noise, as well as current methods for its assessment (chapter 4) – we have proposed a novel, participatory solution to the assessment and mapping of environmental noise and its impact on the quality of life of local communities (section 5.3). The combination of community memories, participatory sensing and social tagging allows for a solution that can be a viable alternative for, or a valuable complement to, conventional methods. In comparison, the NoiseTube approach is cheaper, is entirely based on field observations (rather than simulations), provides a people-centric (rather than location-centric) perspective on exposure, and allows to augment quantitative measurements with qualitative input – regarding sound sources, context, and perception. As opposed to conventional methods which focus on noise alone, our approach allows communities to construct richer representations of soundscapes they are concerned with. Moreover, through the direct involvement of citizens in the assessment and interpretation process, we support the raising of public awareness regarding the issue of environmental noise. The NoiseTube approach can be applied at different, nested scales – by individuals as well as groups, who may or may not contribute to larger, mass sensing efforts. At the group and mass scales we foresee both citizen- and authority-led initiatives for (coordinated) participatory noise mapping campaigns.

Contributes to the tackling of challenge 1.

Contribution III: The NoiseTube system

To underpin the NoiseTube approach we have designed, implemented and iteratively improved a fully functional mobile sensing and community memory system (section 5.4). The NoiseTube system consists of:

Contribution III.a: NoiseTube Community Memory

The NoiseTube Community Memory (CM) handles the processing, storage and aggregation (per user, city or by tags) of data submitted by users of NoiseTube Mobile. Moreover it serves as a Web portal that offers tools to explore, visualise (using maps, charts and tag clouds), analyse, search through and disseminate results. It was the first online community platform aimed at environmental noise and soundscape assessment (section 5.5).

Contribution III.b: NoiseTube Mobile

The NoiseTube Mobile app (chapter 6) allows anyone to turn their mobile phone into a sound level meter (SLM). While not on par in terms of accuracy (see below), it supports various features that are only found in the most expensive SLMs. Besides measuring sound level, users can add qualitative information via social tagging. The (geo-)tagged data can be submitted in real-time to the (or a) NoiseTube CM (instance). However, users are in full control of where and when measurements are made and whether those are geo-tagged and shared. So the app can also serve as a standalone, personal SLM. NoiseTube Mobile is currently available for the Java ME and Android platforms and will soon be released for iOS, at which point it will run on $\pm 85\,\%$ of the smartphones sold in 2011 (see section E.2.2). We have ensured the app can produce data of acceptable quality by closely following SLM standards and devising a calibration/correction method that is superior to related work. By applying software engineering best practices we have facilitated variability and reuse on 3 dimensions: platforms, devices and applications (section 6.7).

Because the NoiseTube system is open source others can benefit from our efforts or make contributions of their own. While ours was not the first, nor the last, attempt at creating a mobile sensing system for sound level measuring and assessment of environmental noise, the NoiseTube system is the most complete solution to date, and in all probability also the most widely used one (section 6.6).

Contributes to the tackling of challenges 1, 4 and 5.

Contribution IV: Multi-scale deployment in practice

As opposed to most mobile sensing systems developed by academics, we have pushed the NoiseTube system well beyond the prototyping and demonstration stage. Since the service was opened up to the general public in May 2009, it has and continues to be used by hundreds, if not thousands, of citizens from across the world. Given the emphasis we have put on designing for multiple sensing scales, it is important to note that we have not just foreseen but also reached these different levels. The NoiseTube Mobile app has been downloaded over 12000 times, which is significantly more than the number of registered NoiseTube CM users, which indicates that many people have indeed chosen to use the app as a personal sensing tool. The efforts of the $\pm$1300 registered NoiseTube users, who are spread over 75 countries and tend to use the app on their own, form a clear example of (uncoordinated) mass sensing. While locally coordinated group sensing efforts have not emerged spontaneously (see below) we have deployed our system at this scale as part of the noise mapping campaigns in Antwerp (section 7.3).

Contributes to the tackling of challenges 1 and 2.

Contribution V: Lessons from usage at the mass sensing scale

By analysing 3 years' worth of data contributions to the NoiseTube CM (sections 5.6 and 5.7) we have been able to draw interesting lessons regarding usage of mobile sensing systems at a mass scale. First of all we found that our user community exhibits a similar degree of *participation inequality* as what is considered [370] to be typical of online communities, which generally do not involve outdoor activities. Second, we found that the contributing portion of our users is too geographically distributed for implicit, uncoordinated collaboration to result in useful noise maps. Third, as it stands, we have not found evidence of lasting, spontaneous (as in, without our direct involvement) coordinated collaborations. This underscores the difficulty of recruitment and the importance of local, "offline" coordination of sensing efforts. Fourth, our data shows that the social tagging feature is being used for the purposes we had foreseen, including the expression of positive sound perceptions. This indicates that, at least in the minds of our users, social tagging indeed brings added value to soundscape assessment.

Contributes to the tackling of challenges 1, 2 and 3.

Contribution VI: Validation of mobile phones as sound level meters

By means of rigorous experiments in the lab and the field, well beyond earlier efforts, we have evaluated the suitability (in terms of precision and accuracy) of mobile phones as SLMs (section 7.2). This has allowed us to confirm the main assumptions that underpin the way measurement correction is implemented in NoiseTube Mobile. We are now confident that model-specific white noise calibration, in combination with our linear interpolation algorithm, enables a level of accuracy close to that of Class 2 SLMs. Accuracy can be further improved by means of device-specific calibration or model-specific calibration based on multiple instances. The validation experiment in the field demonstrated that, when measurements are averaged over longer intervals errors with respect to a Class 2 SLM are negligible, despite the influence of external factors such as wind. This result supports the main hypothesis regarding data quality in participatory sensing systems, namely that the inherent imprecisions and inaccuracies of individual sensors can be compensated through calibration and by statistical reasoning over large datasets with dense spatio-temporal granularity, which participatory sensing allows to collect at low cost.

Contributes to the tackling of challenges 1 and 4.

Contribution VII: Validation of participatory noise mapping

By means of a two coordinated measuring campaigns with volunteering citizens, followed by a user study, we have evaluated – in terms of data quality, usability and organisational aspects – our participatory noise mapping approach in realistic conditions (section 7.3). Through qualitative comparison of the resulting noise maps with an official, simulation-based map, we found strong indications that support the validity (e.g. capturing expected trends) and value (e.g. detection of noise that is underestimated by the official map) of our approach, as well as its complementarity with conventional assessment methods. In other words, "participatory noise mapping works!" [127]. This is the first time participatory noise mapping was validated at this level of scale and realism. More generally (cf. goal 1), this result also demonstrates that, when taken seriously (i.e. using calibrated devices, ensuring spatio-temporal density and overlap, etc.), participatory sensing *can* serve as an alternative or complementary approach to official environmental monitoring efforts.

Contributes to the tackling of challenges 1, 3 and 4.

In the light of the design principles put forth by Elinor Ostrom, and the other guidelines discussed in section 2.3, the contributions listed above represent a significant effort to support the raising of public environmental awareness and especially to facilitate active participation of the public in environmental monitoring (e.g. see principle 4 in box 2.1), by means of CMs, low-cost mobile sensing technology and community science.

8.3 Impact

In the past 4 years we have made substantial efforts to disseminate our work well beyond scholarly publications and venues. Partially because of that, we have been fortunate to receive press coverage on regular occasions. In appendix G we provide a comprehensive overview of scholarly and popular publications, event contributions, and media mentions. Here we give a brief summary of the academic and societal impact of our work:

- as of 2012-04-13 there have been **60 citations** of our papers (h-index = 4);
- we have presented or demonstrated our work at a total of **33 events**, aimed at academic, industrial, governmental, artistic, or general public audiences;
- our research has been the subject of at least **56 mentions in mainstream media** (in 7 countries and 4 languages), concretely there have been 25 mentions in online media, 21 in printed media, 5 appearances on TV, and 5 on the radio;
- the NoiseTube Community Memory [375] has over **1300 registered users** from over 650 cities in 75 counties, spread across all continents except Antarctica;
- the NoiseTube Mobile app has been **downloaded over 12000 times**.

Beyond the conventional academic motivation to "publish rather than perish", our dissemination efforts have served two other purposes. First, because anything "participatory" obviously requires participants, we have sought to attract (potential) users for the NoiseTube system, and potential initiative takers or partners for participatory noise mapping campaigns. Second, motivated by **goal 3**, we intended to contribute to the raising of public, academic and governmental awareness about environmental noise in particular and sustainability challenges in general. In view of the statistics listed above we are confident to say that we have tackled this goal with great success.

Last but not least, it is noteworthy that the work covered in this thesis has effectively introduced a brand new research theme within the VUB's Computer Science department.

8.4 On-going and future work

Here we provide an overview of on-going research efforts that have spun off from the work covered in this dissertation, as well as directions for future research, some of which are the subject of concrete plans or projects. To categorise these we use a slightly altered version of the challenges listed in section 3.4.

Challenge 1: Putting mobile sensing into *policy* practice

In the experiments discussed in chapter 7 we took the perspective of bottom-up, citizen-led initiatives, although most findings are equally relevant to top-down, authority-led initiatives. However, before participatory noise mapping can become an integral part of environmental noise policy it is necessary to consider the specific requirements of authorities. For instance regarding specific data formats, or the general need to integrate with existing systems and practices. We have concrete plans to tackle this policy-oriented challenge in the scope of 2 new research projects.

First there is the *i-SCOPE* project [277], which is funded by European Commission's *ICT Policy Support Programme* and is carried out by a consortium which includes, besides academic and industrial partners, 9 local or regional authorities from 7 countries. The goal of the project is to create an open source toolkit for 3D urban information models (based on the CityGML [394] standard) and associated services, targeted at city administrators and citizens. On top of a joint integrated platform, different policy support services will be developed to address the following three scenarios: improved inclusion and personal mobility of elderly and diversely able citizens; energy dispersion and solar energy potential assessment; and noise mapping and simulation. The latter service will incorporate NoiseTube technology and data collected with NoiseTube. The platform and services will be evaluated in pilot studies organised in the involved cities and regions. We expect this project, which was kicked off in January 2012, to be an ideal context to learn what local authorities expect and how our technology can be adapted to suit their needs.

Second there is the *CART-ASUR* project [311], which is funded by the French national agency for the environment and energy management and involves a consortium of French academic[1] and governmental institutions – specialised in acoustics and cartography – as well as the Paris city council. The objective is to address the limitations of current strategic noise maps as mandated by the EU's Environmental Noise Directive [173]. As discussed in sections 4.3.2.7 and 4.3.2.8, such maps, despite being intended to inform policymakers as wells as the general public, often fail to capture the urban soundscape as it is experienced by citizens. Concretely the project will propose new indicators, and cartographic representations thereof, that better reflect citizens' soundscape perception, thereby making noise maps more readable and useful for policymakers and citizens alike. The project, which is due to start in the course of 2012, involves three stages. First, a mobile app derived from NoiseTube Mobile will be used to let citizens of a Parisian neighbourhood collect data on perceptive (by means of questionnaires) and acoustic (by means of sound level measurements) aspects of the soundscapes at predefined places. Moreover statistics on traffic flows and other known noise sources will be gathered from existing databases. In the second stage, both datasets will be combined to produce

[1] In this project the VUB acts as a subcontractor of Université de Cergy-Pontoise.

a map prototype which can be dynamically adapted to suit different users. In the third stage, semiological and sociological methods will be applied to evaluate and iteratively improve the prototype in terms of usability for different stakeholders. Essentially the CART-ASUR project will develop a new noise mapping approach incorporating elements of both participatory and simulation-based mapping – which are, as argued at various points in this dissertation, inherently complementary.

Challenge 2: *Facilitate* recruiting & retaining users

As noted in section 3.4, work by Reddy et al. [439] indicates that schemes in which contributors are provided with monetary or other incentives can facilitate recruitment and retention in participatory sensing systems. In section 5.3 we argued that this may be especially worthwhile, or perhaps even necessary, in the context of (mass scale) participatory noise mapping campaigns initiated by authorities. So far we have not experimented with such incentives. However, in the scope of the abovementioned CART-ASUR project [311], a budget has been allocated to compensate participants with monetary (calling minutes) or material (smartphones) incentives. Besides helping to ensure that the necessary data is collected (see above), this may provide an opportunity to investigate which incentives work best to motivate or increase compliance for different categories of users.

Another possible avenue for future efforts to tackle this challenge is the introduction of elements of competition and gaming in the NoiseTube system. As mentioned in section 6.6, Schweizer et al. have introduced such elements in their *NoiseMap* app [466, 467], which could be a source of inspiration here.

Challenge 3: *Better support for* collaboration & coordination

Up to now we have approached the challenge of collaboration and coordination from an organisational and human perspective, focusing especially on the group sensing scale. However, especially at a mass scale participatory environmental monitoring campaigns may benefit from automated, or at least technically supported, efforts to encourage explicit collaboration and coordination thereof (see section 3.4). In a 2010 workshop paper [499] we already hinted on the possibility of introducing a central, a (semi)-automatic coordination subsystem that analyses users' mobility patterns and sends out route suggestions to fill coverage gaps or avoid double work. We have yet to experiment with such a solution. However, table 8.1 below lists further ideas for tackling this challenge. We consider two dimensions: timing and coordination. The first differentiates cases where the division/organisation of work is planned in advance, and those where collaboration opportunities are spotted and grasped in real-time. The second draws a distinction based on whether collaboration is locally (LC) or centrally (CC) coordinated.

Especially in mass sensing, but possibly also in group sensing, we envision work to be divided/organised by means of "push" messages, sent from the server to

		Timing	
		Beforehand / Planned	**Real-time / Spontaneous**
Coordination	**Local (LC)** Locally coordinated by users	**Online discussion** (mass & group scale): forum, wiki, mailing list, social network, … **Offline discussion** (group scale only): face-to-face meetings **Client-to-client push** (especially mass scale)	**Client-to-client push** (especially mass scale) Example: *"Hi Alice, I'm Bob, a fellow user of system X. You seem to be in the same neighbourhood and maybe we can divide the work. How about today I measure in streets A, B & C, while you take care of streets D, E & F?"*
	Central (CC) Centrally coordinated by the system and/or its owners/operators	**Data collection "calls"** (especially mass scale): posted online, or as a **server-to-client push** (to all or some subset of users) Example: *"Request: We need 5 users to measure in streets A, B & C between time 1 & 2, any takers?"*	**Server-to-client push** (especially mass scale) Example: *"Dear Bob, you seem to be in neighbourhood M at an interesting time, would you mind collecting data in streets A, B & C? Thanks!"*

Table 8.1: Ideas for the stimulation/facilitation of explicit collaboration among users of participatory mass or group sensing systems

clients (CC), or between clients (LC), which invite users to take on certain tasks. For this to work in real-time, at least some users must be willing to, more or less constantly, disclose their location[2] to the server (CC) and/or other users (LC)[3]. In case of LC, the client app could include a dynamic map showing the whereabouts/activity of users in the immediate neighbourhood, and let users contact one another. In case of CC, the spotting of collaboration opportunities and sending of invitations may happen manually (i.e. via human intervention[4]), semi- or fully automatically.

Challenge 4: *Further assessment and improvement of* data quality

To investigate further the accuracy and precision of mobile phones as SLMs we could conduct additional field validations experiments in which measurements made on phones are compared with those made by nearby SLMs (as in section 7.2.5). This could provide further insights in the accuracy under different real-world conditions. Taking inspiration from the work of Santini et al. [459] it may also be useful to conduct laboratory experiments with pre-recorded sounds. This would allow us to assess measurement accuracy for various typical urban sounds in a reproducible manner, without external influences such as wind. As noted in section 7.3.6, one way to assess further the quality of participatory noise maps could be to apply descriptive statistics techniques. Another would be collect data over 24 hour periods in order to make L_{den} maps that can be compared directly with official ones.

Avenues that can be pursued to (potentially) improve the quality of participatory noise maps include (semi-)automatic data clean-up algorithms[5], reputation systems

[2] Possibly obfuscated.

[3] Clearly, such solutions require the matters of privacy and context-awareness to be considered as well (see challenges 6 and 7).

[4] By the system's owners and/or designated operators (possibly members of the user community).

[5] Which could possibly solve problems such the cluster of indoor measurements discussed in section 7.3.5.

such as proposed in [261], or perhaps "Wikipedia-style" community-driven reviewing of data contributions at the CM level.

Challenge 5: Reusable *Web-based* components

Reusability and platform-independence have been an important concern in the design, implementation and refactoring of NoiseTube Mobile (see section 6.5.6 and contribution III.b). Nevertheless the current, fragmented landscape of smartphone platforms (see appendix E) makes it difficult to target all popular devices using the same code – for instance, the new iOS variant of our app has been written from scratch because this platform does not support Java. Therefore we are looking for new ways to make mobile sensing apps more platform-independent. As the line between locally installed and Web applications continues to blur, and powerful new APIs are being drafted as part of (or related to) the emerging HTML5 standard [605], there is hope that soon it will become possible to build advanced mobile sensing apps such as ours using only platform-independent Web-technologies.

To explore some initial possibilities we let one of our bachelor students implement a subset of the NoiseTube Mobile functionality using a Web-based, cross-platform mobile application framework called *PhoneGap* [373].

Challenge 6: *Improved* privacy

While the NoiseTube system includes a few simple, common sense features to increase user trust and control (see sections 5.3 and 6.3.6.2), we have initially refrained from developing more advanced solutions. However, thanks to a recent collaboration with cryptography specialists we have been able to tackle concerns over the disclosure of personal location traces in an entirely new way. The solution consists of two parts. First, we introduce the notion of personal software agents representing NoiseTube users. Each agent runs on a system (remotely) controlled by the user – separately from NoiseTube Mobile and the CM. Typically agents would be hosted on a commercial cloud computing infrastructure. The agents serve two main purposes: providing secure, encrypted storage for the user's sensor data and representing the user when responding to data aggregation requests. Up to here the solution is similar to the notion of a *personal data vault*, proposed in [153]. However the second part of the solution goes well beyond that. Using a Web application, campaign coordinators can send out a data aggregation request to a set of agents. This initiates a distributed computation process in which the agents collaborate to produce an aggregated noise map in an entirely *privacy-preserving* fashion, through the use of a *homomorphic encryption* scheme. The system is able to generate identical grid-based noise maps (along with statistical metadata) without any personal location tracings being disclosed to anyone, including the cloud service provider(s) on which the agents run. This work, the results of which have yet to be integrated in the NoiseTube system, is described in a co-authored paper [135], which is due to be presented at the COMPSAC 2012 conference.

Challenge 7: Context-awareness

Apart from some simple features to increase transparency[6], NoiseTube Mobile does not provide context-aware features. However, we have some (partially tested) ideas for new features that would leverage context-awareness.

In a 2009 position paper [547] we argued that participatory sensing could benefit from apps that are aware of nearby devices (and their users) and can engage in seamless, ad-hoc communication with those. That could, for example, enable new ways to collaborate and share resources in a peer-to-peer (i.e. server-less) manner. In 2011 an initial investigation of this potential was conducted in the scope of a master's thesis [39] prepared, under our guidance, by Sander Bartholomees. The main contribution of this work is a framework that allows mobile sensing apps to dynamically detect the presence of other devices – "peers" as well as external Bluetooth-connected sensors such as as [470] and [371] – and provide and/or subscribe to peer-to-peer services. Examples of such services are the sharing of sensor data (e.g. when a device has sensors that are not available on other, or when readings of multiple sensors of the same type are to be combined), processing of data collected on other devices (i.e. sharing CPU capacity), or relaying of data (e.g. to allow devices without Internet connectivity to send data to a server through another nearby, Internet-connected device). This framework was implemented in *AmbientTalk* [14, 548], a programming language that is specifically designed to simplify the development of distributed, context-aware applications operating over mobile ad hoc networks (e.g. Wi-Fi or Bluetooth)[7]. With his framework Sander created an experimental extension of NoiseTube Mobile for Android and tried out 3 validation cases (data relaying, GPS sharing, and sharing of SPL measurements).

Another possibility could be to make NoiseTube Mobile aware of the so-called *phone context* [310], and more specifically the device's position with respect to the carrier (e.g. in a pocket or bag, in the hand, on the hip or arm). In [350] Miluzzo et al. propose a machine learning algorithm that makes this possible by relying on multiple sensors (e.g. microphone, camera or light sensor, accelerometer, gyroscope and compass). Using such an algorithm NoiseTube Mobile could automatically pause measuring when the user puts his/her phone away (when the user forgets to do so manually) and resume measuring when the phone is taken back out – thereby avoiding unreliable sound level measurements.

[6] For instance automatically pausing measuring when the app is put in the background or when the user makes or receives phone calls (see section 6.3.6.1).

[7] AmbientTalk is developed at the VUB's Software Languages lab, which also hosts the BrusSense team. Hence this master thesis was an excellent opportunity to marry two research tracks pursued in the lab (i.e. ambient oriented programming and mobile sensing).

Challenge 8: Continuous sensing vs. Autonomy

Because NoiseTube users can themselves decide when and for how long they make measurements, there was no immediate need to develop technical solutions to reduce energy consumption (and thereby increase device autonomy). Nevertheless it could be worthwhile to evaluate some potential solutions proposed in literature, most of which rely on duty-cycling – the interleaving of sensor sampling, computation and communication with periods of inactivity. Some possible starting points or sources of inspiration are [328, 359, 562, 607].

Challenge 9: Sensing & Learning more

In view of contribution I it should be possible to apply our general approach to other commons challenges besides environmental noise. Within the Innoviris-funded *BrusSense* project [124] we aim to do precisely that. Concretely the project calls for a broader focus in which air pollution and urban microclimates are monitored in parallel with environmental noise. We do this by building on the concepts and technologies developed in the NoiseTube project and discussed in this dissertation. To monitor atmospheric pollutants and weather conditions we use wearable or portable sensors [371, 470] which communicate with mobile phones via Bluetooth. On the phones we use an app, extended from NoiseTube Mobile, that combines locally collected sound level measurements with data collected by the external sensors. This once more confirms the application variability allowed by the architecture of NoiseTube Mobile (see section 6.5.6 and contribution III.b). A first measurement campaign, targeting the typical tourist route in Brussels, took place in April 2012. Preliminary results can be seen in [67]. These new sensor types will necessitate new, and arguably more complex, assessments of data quality.

8.5 Closing remarks

To wrap up this dissertation, we once more stress that the contributions presented here, in combination with our extensive dissemination efforts, have allowed us to achieve all three of the goals we set out to achieve in section 1.3. Moreover, as demonstrated above, the work covered in this dissertation is a fruitful basis for interesting – as well as, dare we say, exciting – on-going and future research efforts.

Appendices

Appendix A

All about sound

A.1 Introduction

The goal of this appendix is to explain what sound *is* and how it is *perceived* (i.e. heard), *captured* (i.e. recorded or transmitted) and *measured*. With this we intend to provide the reader with the essential insights, as well as adequate background information, for a proper understanding of the discussion of (environmental) noise in chapter 4 and of our NoiseTube system in chapters 5 to 7.

First we introduce the physical phenomenon of sound and some relevant properties and measures in section A.2. Next, section A.3 treats how humans perceive sounds and what influences their *loudness*. Then, section A.4 explains how sound is represented with analogue and digital audio signals, which can be stored or transmitted in various formats. This is meant to support the comprehension of the technical aspects of sound level meters and of our NoiseTube Mobile application, which is discussed in chapter 6. Finally, section A.5 explains how sound is measured using sound level meters and dosemeters.

The content of sections A.2, A.3 and A.5 was assembled and cross-checked using a variety of sources. Besides those cited in the text, notable ones are the *Little Red Book of Acoustics* by Watson & Downey [564], the AIHA *Noise Manual* edited by Berger et al. [50], the *Cambridge IGCSE Physics Coursebook* by Sang [457] and the syllabus of the VUB course on acoustics and noise nuisance by Vanlanduit & Van Overmeire [550]. Furthermore some online sources were consulted, notably the *Physics Classroom* [251] and acoustics-related entries in the *Wikipedia encyclopaedia* [598]. The content of section A.4 is based on personal expertise in the matter of digital audio, complemented with two additional sources, *Principles of Digital Audio* by Pohlmann [424] and audio-related entries in the *Wikipedia encyclopaedia* [598].

A.2 Physics of Sound

In this section we explain what sound is and discuss a number of important acoustic properties and measures. This account is by no means exhaustive. The goal is only to provide readers without a background in acoustics with sufficient information to support the understanding of the next sections and the main chapters. For a more comprehensive introduction to acoustics we kindly refer the reader to the abovementioned sources.

A.2.1 Sound Waves

Simply put, *sound* is nothing but the vibrations, also called *oscillations*, of a *medium*, typically the air that makes up the atmosphere around us. A *sound source*, such as a tuning fork or a loudspeaker, produces sound by vibrating. The object's vibrations are picked up by the surrounding medium (gas, liquid or solid), causing them to travel through the medium as a *mechanical*, or *material*, *wave* of forward and backward moving molecules. Along the way such *sound waves* can be reflected, refracted, or attenuated by the medium or any obstacles in it. When such a wave reaches us and is detectable – or rather, *audible* – by our ears, we call it a sound.

To gain more insight into the propagation of sound waves in air, imagine the following experiment. Suppose we take a tuning fork and set it to vibrate, by striking it against a surface or with an object. The fork's outer ends, called *tines*, vibrate back and forth causing the surrounding air to vibrate along. If we place the fork in a central position in an open space, sound waves spherically radiate away from it, as shown in figure A.1.

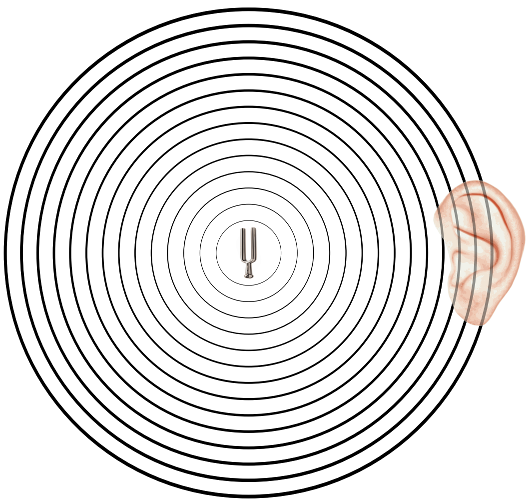


Figure A.1: Sound waves generated by a tuning fork, propagating in all directions

Now suppose we have a tube which is open on both ends. We hold the tube close to the vibrating tuning fork with one opening facing it. Figure A.2 illustrates what happens.

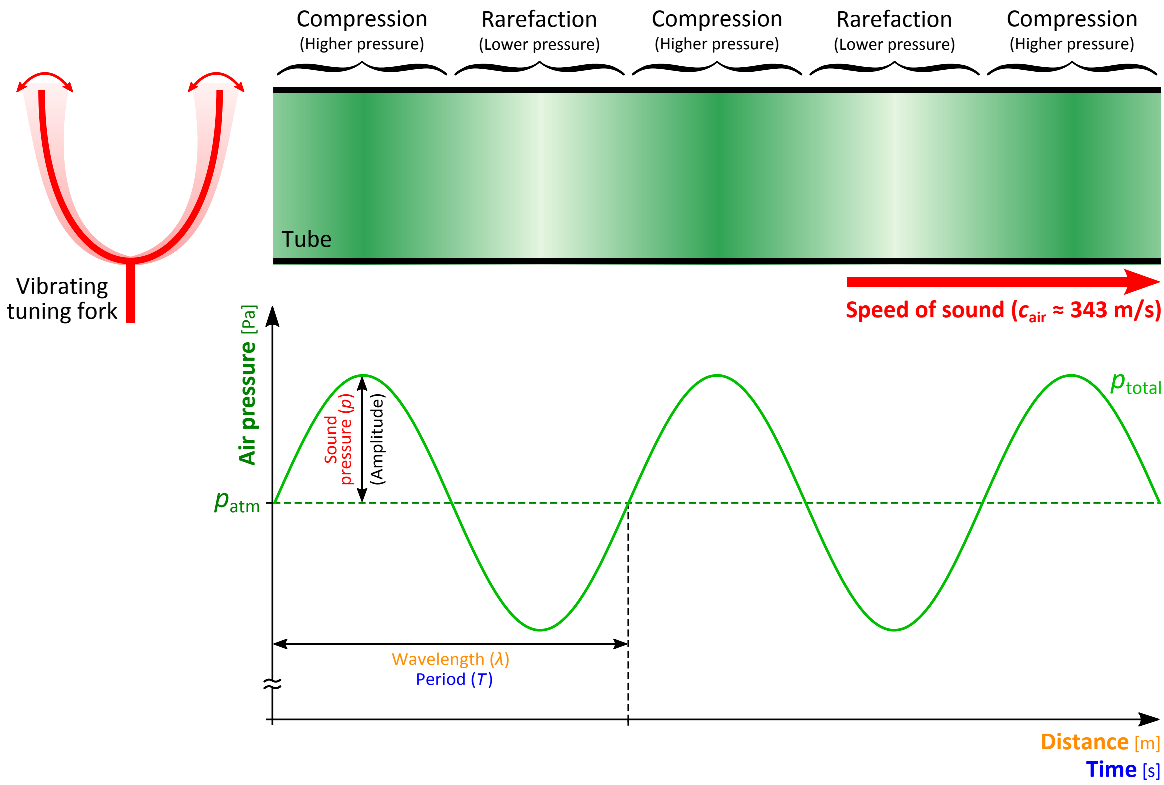


Figure A.2: Propagation of a sound wave through the air in an open tube

Each outward extension of a tine pushes neighbouring air molecules away from the fork, causing them to push on their neighbours (and so on), creating a *compression*, an area where the molecules are pressed together. Each inward retraction of a tine creates a low-pressure area, allowing air molecules to move back towards the fork and thus creating a *rarefaction*, an area where the molecules are spread apart. So as the air molecules are pushed back and forth by the vibrating fork, local and temporal changes in air pressure occur in the tube. If we assume that our experiment takes place at an ambient atmospheric pressure of p_{atm}, then the local air pressure in compressions and rarefactions is respectively higher and lower than p_{atm}. Sound is thus a *pressure wave*, an alternating pattern of higher- and lower-pressure regions travelling through a medium. We now study this pressure wave in the space domain and in the time domain.

To look at the space domain, suppose we measure the local air pressure (p_{total}) in different locations in the tube at the same point in time, and then plot these values in function of the distance from the tuning fork. The result is shown on the chart in figure A.2, with the horizontal axis representing distance[1].

[1] With each plotted value corresponding to the location directly above it in drawing of the tube.

Because the sound produced by our tuning fork approximates a so-called *pure tone*, the plotted waveform follows a sinusoidal pattern. The length of one complete cycle of the wave – for example measured from the beginning of one compression to the next, as indicated on the chart – is called the *wavelength* and is denoted by λ.

To study the time domain, suppose we measure the local air pressure (p_{total}) at different points in time at a single, fixed location in the tube, and then plot these values in function of their measurement time. If the measurement site is at a distance of $z \cdot \lambda$ away from the tuning fork, with $x \in \mathbb{Z}$, we get exactly the same waveform as before. The result is shown on the chart in figure A.2, but now the horizontal axis represents time.

The passage of the pure tone sound wave through the measurement site thus causes air pressure fluctuations which over time follow the same sinusoidal pattern that appears in the space domain. At one moment a high pressure is measured, corresponding to the arrival of a compression at the site. At the next a normal pressure ($= p_{\text{atm}}$) is measured. And then a low pressure is measured, corresponding to the arrival of a rarefaction at the site. Then a normal pressure is measured again and the cycle repeats itself. The duration of one complete cycle of the wave is its *period* and is denoted by T. The wavelength we discussed above is thus the distance the sound travels during one period, as defined by the *speed of sound*, denoted by c, for the medium being traversed. For dry air[2] at 20 °C this is about 343 m/s. The reciprocal of a wave's period is its *frequency*, denoted by f, the number of cycles per unit of time. Equations A.1 and A.2 summarise the relations between these parameters:

$$f = \frac{1}{T} \tag{A.1}$$

$$c = \frac{\lambda}{T} = \lambda \cdot f \tag{A.2}$$

The SI unit for frequency is the hertz (Hz)[3]. Pure tones, which consist of only a single frequency, are usually generated artificially – even our tuning fork does not produce a "perfect" pure tone. Most other sounds around us are composed of multiple frequencies and hence do not have a sinusoidal waveform, such as in chart A.1. The frequencies present in sound waves are related to *pitch*, which is the subjective measure of how *high* or *low*(-*pitched*) particular sounds appear to our ears and brains (see section A.3.3).

[2] The speed of sound in air (c_{air}) varies with temperature and (to a lesser degree) humidity. An approximation of c_{air} at a given temperature t (in °C) can be calculated as: $c_{\text{air},t} \approx 331.3 + 0.606t$ [m/s].

[3] $1\,\text{Hz} = \frac{1}{\text{s}}$ (cycles per second).

A.2.2 Sound Pressure

The chart in figure A.2 also indicated the wave's *amplitude*[4]. For a sound wave this is the *sound pressure* or acoustic pressure, denoted by p. It is the *deviation* from the ambient air pressure occurring at each location, when looking at the space domain, or at each moment, when studying the time domain. In the latter case we call it the *instantaneous sound pressure*[5], and for any instant t it is given by:

$$p(t) = p_{\text{total}}(t) - p_{\text{atm}} \quad [\text{Pa}] \tag{A.3}$$

The relation between p, p_{atm} and p_{total} is illustrated once more by chart A.1. The SI unit for sound pressure is the pascal (Pa)[6]. Compared to the average sea-level atmospheric pressure of 101325 Pa, the pressure deviations caused by a sound wave – i.e. the sound pressure – are very small (see table A.1 on page 271 for some examples). Note that the instantaneous sound pressure can be negative, as illustrated by instant t_x in the chart.

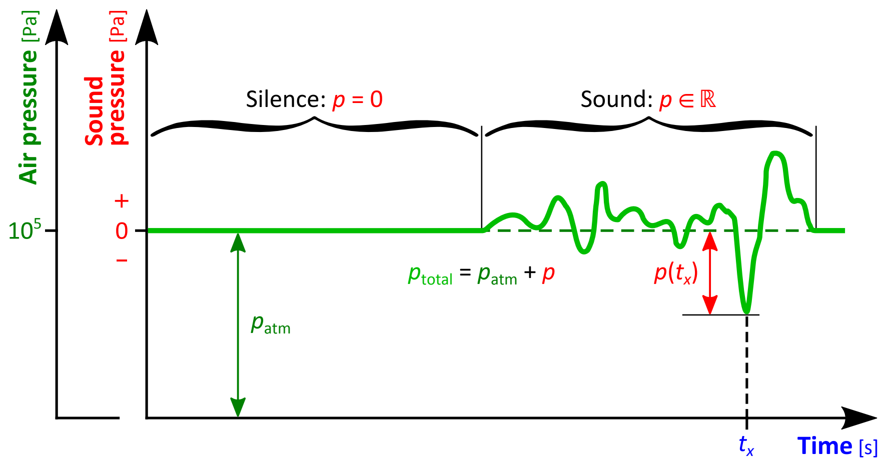


Chart A.1: Atmospheric pressure and (instantaneous) sound pressure

Sound pressure is related to *loudness*, which is the subjective measure of how *loud* particular sounds appear to us (see section A.3). However, when we perceive any sound as having a constant loudness this does not at all mean that the sound pressure is constant over time. As the sound wave completes a full cycle during each period, the instantaneous sound pressure (i.e. the wave's amplitude) fluctuates continuously and is negative roughly half of the time. Because of this instantaneous sound pressure is not a good measure to describe the effect of sounds and neither is its arithmetic mean (which could be zero). Also, human hearing is fairly insensitive to the minimum and maximum sound pressure (i.e. positive and negative peak amplitudes). However, we *are* sensitive to the

4 In fact, the chart in figure A.2 indicates the *peak* amplitude.

5 While it is not a commonly done, one could define a space domain counterpart, the *"local sound pressure"*, which for any distance d from the source is given by: $p(d) = p_{\text{total}}(d) - p_{\text{atm}}$ [Pa].

6 $1\,\text{Pa} = 1\,\frac{\text{N}}{\text{m}^2}$ (Newtons per square metre).

energy conveyed by sound waves[7]. This has led to the use of the *effective sound pressure* (ESP) as a means to quantify the strength of sound. This measure is denoted by p_{eff} (sometimes p_{rms}) and is defined as the *root-mean-square* (RMS) of the time-varying sound pressure over a certain time interval. For a given averaging time[8] T it is calculated as follows:

$$p_{\text{eff}} = p_{\text{rms}} = \sqrt{\frac{1}{T}\int_0^T p^2(t)\,\mathrm{d}t} \quad [\text{Pa}] \tag{A.4}$$

The term under the square root sign is the *mean-square* sound pressure: the squared instantaneous sound pressure, $p^2(t)$, averaged over an interval with duration T.

A.2.3 Sound Pressure Level

Human hearing can perceive sound pressure from about 20 µPa (micropascals) to 200 Pa. This huge range – spanning 7 orders of magnitude – has led to the definition of a more practical, logarithmic measure called *sound pressure level* (SPL), often shortened to just *sound level*. This measure is denoted by L_p and uses the *decibel* (dB)[9] notation to express the ratio of the effective sound pressure to the *reference sound pressure*, denoted by p_0 (sometimes p_{ref}), which is taken as 20 µPa for measurements in air[10]. For a given ESP value p_{eff} the SPL L_p is calculated[11] as follows:

$$\begin{aligned} L_p &= 10 \cdot \log_{10}\left(\frac{{p_{\text{eff}}}^2}{{p_0}^2}\right) \quad [\text{dB}] \\ &= 20 \cdot \log_{10}\left(\frac{p_{\text{eff}}}{p_0}\right) \quad [\text{dB}] \end{aligned} \qquad \text{With: } p_0 = 20\,\mu\text{Pa} = 2{\times}10^{-5}\,\text{Pa} \tag{A.5}$$

Table A.1 lists SPL values from 0 to 140 dB, in steps of 10 dB, with the corresponding ESP values. To facilitate the interpretation of SPL values we added examples of sources which produce sounds at particular levels, or places where the ambient (background) sound can be expected to fluctuate about certain levels. Note that in principle SPL values can be negative, namely for ESP values below the reference sound pressure[12].

[7] The energy of sound waves per time unit is the *sound power* (expressed in watts), while *sound intensity* is defined as sound power per unit area (expressed in watts per m^2). In free-field conditions both sound power and sound intensity are proportional to $p_{\text{eff}}^2 = \frac{1}{T}\int_0^T p^2(t)\,\mathrm{d}t$ (the *mean-square* sound pressure).

[8] Interval duration T in equation A.4 should not be confused with wave period T in equations A.1 and A.2.

[9] The *decibel* is a dimensionless quantity based on the logarithm of the ratio of two quantities related to power. The *level* of two quantities A and B is defined as: $L = 10\log_{10}\left(\frac{A}{B}\right)$ [dB]. Besides SPL, decibels are also used with other acoustic measures (e.g. *sound power level* and *sound intensity level*; which are outside the scope of our discussion), as well as in non-acoustic contexts.

[10] The value of p_0 was selected to approximately equal the threshold of normal human hearing at 1000 Hz.

[11] The ESP represented by a given SPL value can be found by reversing equation A.5: $p_{\text{eff}} = p_0 \cdot 10^{\frac{L_p}{20}}$ [Pa].

[12] Such sounds are generally too silent to be heard by humans, except for a narrow band of frequencies which may be audible to slightly below 0 dB for young people with undamaged hearing (see section A.3).

<table>
<tr><th>Sound Pressure Level (L_p)</th><th>Effective Sound Pressure (p_{eff})</th><th colspan="3">Examples (sound sources or places)</th><th>Effects on hearing</th></tr>
<tr><td>140 dB</td><td>200 Pa</td><td>Gunshot at close range</td><td rowspan="2">Jet aircraft at < 50 m (during takeoff)</td><td>Fireworks at < 10 m</td><td>≥ 140 dB: Immediate hearing damage</td></tr>
<tr><td>130 dB</td><td>≈ 63.2 Pa</td><td rowspan="2">Alarm siren at 1 m</td><td rowspan="2">Thunderclap (close to ground stroke)</td><td>≈ 125 dB: Threshold of pain (may vary)</td></tr>
<tr><td>120 dB</td><td>20 Pa</td><td>Car horn at 1 m</td><td rowspan="2">≥ 120 dB: Hearing damage possible (short-term exposure)</td></tr>
<tr><td>110 dB</td><td>≈ 6.32 Pa</td><td>Yelling directly in one's ear</td><td rowspan="2">Disco; amplified music concert</td><td rowspan="3">Factory hall with heavy machinery</td></tr>
<tr><td>100 dB</td><td>2 Pa</td><td>Helicopter at 30 m</td><td></td></tr>
<tr><td>90 dB</td><td>≈ 6.32×10^{-1} Pa</td><td>Heavy truck at 10 m</td><td rowspan="2">Traffic on a busy road at 10 m</td><td rowspan="2">≥ 85 dB: Hearing damage possible (long-term exposure)</td></tr>
<tr><td>80 dB</td><td>2×10^{-1} Pa</td><td>Pneumatic drill at 15 m</td><td rowspan="3">Passenger car at 10 m</td></tr>
<tr><td>70 dB</td><td>≈ 6.32×10^{-2} Pa</td><td colspan="2">Vacuum cleaner at 1 m</td><td></td></tr>
<tr><td>60 dB</td><td>2×10^{-2} Pa</td><td>TV set (at home level) at 1 m</td><td rowspan="3">Conversational speech at 1 m</td><td></td></tr>
<tr><td>50 dB</td><td>≈ 6.32×10^{-3} Pa</td><td>Dish washer at 1 m</td><td rowspan="2">Moderate rainfall; average home</td><td></td></tr>
<tr><td>40 dB</td><td>2×10^{-3} Pa</td><td>Quiet library</td><td></td></tr>
<tr><td>30 dB</td><td>≈ 6.32×10^{-4} Pa</td><td colspan="2">Quiet rural area</td><td rowspan="2">Very calm room; whisper at < 5 m</td><td></td></tr>
<tr><td>20 dB</td><td>2×10^{-4} Pa</td><td colspan="2">Soundproof room (e.g. TV or music studio)</td><td></td></tr>
<tr><td>10 dB</td><td>≈ 6.32×10^{-5} Pa</td><td colspan="3">Leafs rustling in the distance; calm breathing</td><td>Barely audible</td></tr>
<tr><td>0 dB</td><td>2×10^{-5} Pa</td><td colspan="3"></td><td>Auditory threshold at 1 kHz</td></tr>
</table>

Table A.1: Sound pressure levels with corresponding effective sound pressures, practical examples and hearing effects

As a consequence of the decibel notation, multiplications (or divisions) of sound pressure correspond to additions (or subtractions) of sound pressure level, and vice-versa:

$$p'_{\text{eff}} = \alpha \overset{\times}{\underset{\div}{}} p_{\text{eff}} \overset{\text{(A.5)}}{\Longleftrightarrow} L'_p = L_p \pm 20 \cdot \log_{10}(\alpha) \quad \text{[dB]} \tag{A.6}$$

So when the ESP of a sound is doubled or halved[13], the corresponding SPL respectively increases or decreases by about 6 dB (since $20 \cdot \log_{10}(2) \approx 6.02$); and when the ESP goes up or down tenfold, the SPL respectively increases or decreases by exactly 20 dB. Chart A.2 shows additional “rule of thumb” tricks for (approximate) conversions.

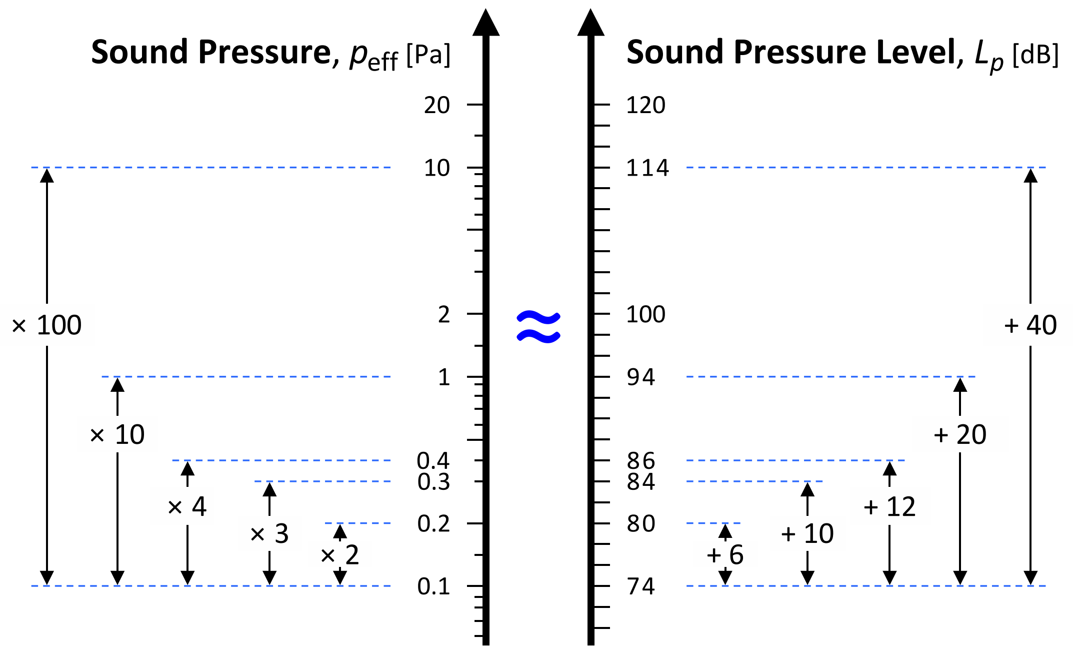


Chart A.2: Relation between sound pressure and sound pressure level

[13] Which does not at all mean that the sound will be perceived twice or half as *loud* (see section A.3.4).

A.2.4 Distance attenuation

As we all know, sounds become quieter the further one moves away from their source[14]. When talking about the sound pressure (level) produced by a particular sound source it is therefore important to always specify how far away from the source the value was measured. To formalise this property of sound waves, we must differentiate between two theoretical types of sound sources: *point sources* and *line sources*.

A.2.4.1 Point sources

In this case the sound radiates as a spherical wave out of a single point is space. Examples of sources behaving roughly as point sources are a tuning fork or a single car. For point sources, the distance attenuation is formalised by the $^1/_r$-*law* or *distance law*, which states that, in free-field conditions (i.e. without any obstacles blocking the sound path), the sound pressure p is inverse-proportional to the distance r from the sound source:

$$p \propto \frac{1}{r} \tag{A.7}$$

So for a single point source, the ESP value p_2 which would be measured at a distance r_2 can be calculated from another ESP value p_1 measured at a known distance r_1 as follows:

$$p_2 = p_1 \cdot \frac{r_1}{r_2} \quad \text{[Pa]} \tag{A.8}$$

By applying equation A.5 we find that, based on an SPL L_{p_1} measured at a distance r_1, the SPL L_{p_2} at a distance r_2, can be calculated as:

$$L_{p_2} = L_{p_1} + 20\log_{10}\left(\frac{r_1}{r_2}\right) \quad \text{[dB]} \tag{A.9}$$

For example, if a TV set causes an SPL of 60 dB at 1 m the SPL is $\approx$ 54 dB at 2 m and $\approx$ 48 dB at 4 m. A practical thing to remember is that doubling the distance from a point source decreases the SPL by about 6 dB.

A.2.4.2 Line sources

In this case the sound radiates as a cylindrical wave out of a single line in space. A typical example of a source that is broadly similar to a line source is a busy road. For line sources, again in free-field conditions, the sound pressure p is inversely proportional

[14] For the sake of simplicity we have ignored this fact in our discussion of the space domain behaviour of sound waves in section A.2.1.

to the square root of the distance r from the sound source:

$$p \propto \frac{1}{\sqrt{r}} \tag{A.10}$$

So for a single line source, the ESP value p_2 which would be measured at a distance r_2 can be calculated from another ESP value p_1 measured at a known distance r_1 as follows:

$$p_2 = p_1 \cdot \sqrt{\frac{r_1}{r_2}} \quad [\text{Pa}] \tag{A.11}$$

By applying equation A.5 we find that the corresponding SPL can be calculated as:

$$L_{p_2} = L_{p_1} + 10 \log_{10}\left(\frac{r_1}{r_2}\right) \quad [\text{dB}] \tag{A.12}$$

For example, suppose we measure an SPL of 90 dB at 10 m away from a highway, we would still measure $\approx$ 87 dB at 20 m and $\approx$ 84 dB at 40 m. In other words, per doubling of the distance from a line source the SPL decreases by about 3 dB.

A.2.5 Addition of sounds

In practical situations we are rarely exposed to the sound of a single source, rather what we usually hear is a mix of sounds originating from different sources. Therefore it is useful to know how multiple sources with individual sound pressures (levels) combine into a sound pressure (level) measured at a given location.

When p_x is the ESP measured from a source x (in isolation) at a particular location, and p_y is the ESP measured from a source y (again in isolation) at that same location, then the ESP measured at that location when both sources are active would be:

$$p_{(x+y)} = \sqrt{{p_x}^2 + {p_y}^2} \quad [\text{Pa}] \tag{A.13}$$

By applying equation A.5 we find that the corresponding SPL is to be calculated as the *logarithmic addition* (which we will denote with $\oplus$) of the individual SPLs L_{p_x} and L_{p_y}:

$$L_{p_{(x+y)}} = L_{p_x} \oplus L_{p_y} = 10 \cdot \log_{10}\left(10^{\frac{L_{p_x}}{10}} + 10^{\frac{L_{p_y}}{10}}\right) \quad [\text{dB}] \tag{A.14}$$

For example, suppose we measure an SPL of 90 dB coming from a truck at 10 m away, then the arrival of a second, similar truck, at the same distance, would cause the SPL to rise to $\approx$ 93 dB. So, per doubling of the amount of equal sources at equal distance, the SPL increases by about 3 dB.

By generalising equation A.14 for n sources we get:

$$L_{\sum_{i=0}^{n} p_i} = L_{p_1} \oplus L_{p_2} \oplus \ldots \oplus L_{p_n} = 10 \cdot \log_{10}\left(\sum_{i=1}^{n} 10^{\frac{L_{p_i}}{10}}\right) \quad [\text{dB}] \tag{A.15}$$

For example, suppose we have a vacuum cleaner, a TV set and a dish washer, which in isolation respectively produce an SPL of 70, 60 and 50 dB at 1 m, then the SPL measured at 1 m when the appliances are simultaneously active would be ≈ 70.5 dB.

A.2.6 Averaging of sound pressure levels

Because decibel is a logarithmic unit, average sound pressure levels are typically not computed as an arithmetic mean (although that is not necessarily wrong [50, 564]). Instead, the average of a series of SPL measurements, L_{p_1} to L_{p_n}, is usually taken as[15]:

$$\overline{L_p} = 10 \cdot \log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} 10^{\frac{L_{p_i}}{10}}\right) \quad [\text{dB}] \tag{A.16}$$

For example, the average SPL of 50, 60 and 70 dB is ≈ 65.7 dB.

A.3 Hearing: Perception of sound

To interpret measurements of physical properties of sound, it is useful to have a basic understanding of the anatomy of the human ear, the range of our hearing and some of the psychological aspects of sound perception.

A.3.1 Anatomy of the human ear

As mentioned in section A.2, the deviations in air pressure caused by sound waves are very small compared to the atmospheric pressure. Still we are able to sense them because the human ear, the anatomy of which is illustrated in figure A.3, cancels out the ambient air pressure via the *eustachian tube*, which connects the throat to the inside of the ear. When we yawn or swallow this tube opens, equalising the pressure on both sides of the eardrum or *tympanic membrane*. In this equalised state our ears can sense the small air pressure fluctuations due to sound waves. As sound waves impact on our eardrums they make them vibrate. On the inside of the eardrum, three little bones, the *auditory ossicles*,

[15] This is the same as taking the RMS of the corresponding ESPs and converting the result back to SPL.

transfer the vibrations of the eardrum to the *cochlea*, a fluid-filled organ in the inner ear. Here, sensory hair cells pick up vibrations in the fluid and translate them into signals which are sent to the brain via the *cochlear nerve*. Our brain then interprets these signals and thereby evokes the sensation of hearing.

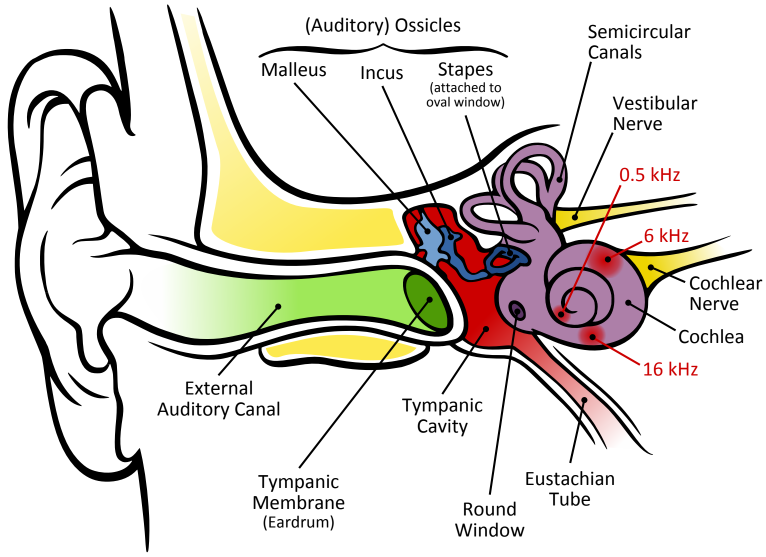


Figure A.3: Anatomy of the human ear, with a mapping of cochlear areas to frequencies

A.3.2 Human hearing range

As we learned in section A.2, sound waves can contain various frequencies and can vary in amplitude. To be audible by humans sounds must contain frequencies within our *frequency range* and have a sound pressure level above our *auditory threshold* for the frequencies in question. In chart A.3 the typical human *hearing range* is shown as an area in the frequency-SPL plane, with frequency being represented on a logarithmic scale and sound pressure level (itself a logarithmic measure) on a linear scale.

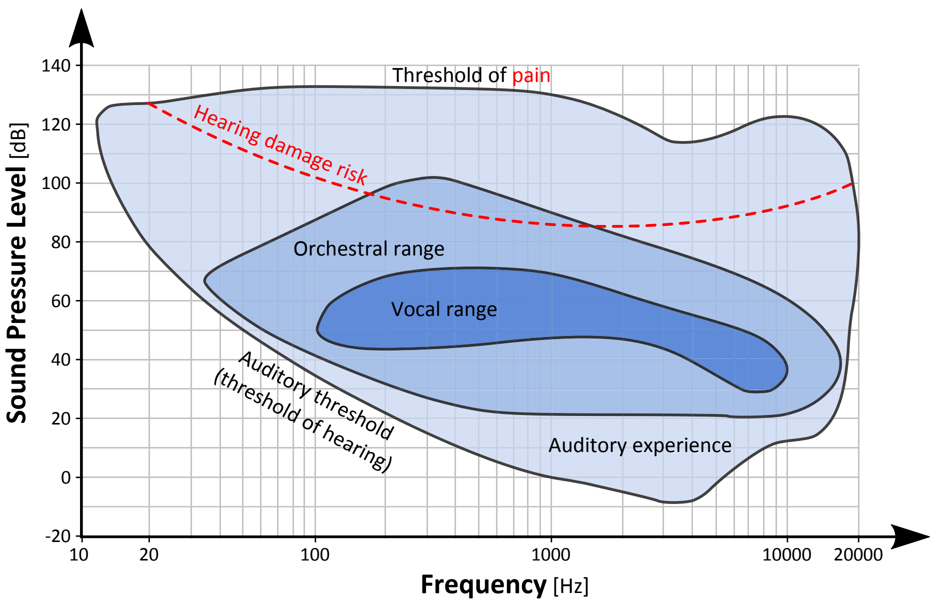


Chart A.3: Typical range of human hearing in terms of sound pressure level and frequency

The range of human-audible frequencies is generally taken to be from 20 Hz to 20 kHz, although there is considerable variation between individuals, especially at the high end of the range. Also, sensitivity to high frequencies gradually declines as we get older. The range of human-perceivable amplitude levels, the *dynamic range* of our ears, spans roughly 130–140 dB for young people with undamaged hearing. It is bounded from below by the auditory threshold, which depends on the frequency. Our hearing is most sensitive in the frequency band between 2 and 5 kHz and the threshold reaches its lowest point at about 3.5 kHz, where sounds as low as -10 dB may be audible. The dynamic range is bounded from above by the *threshold of pain*, above which sounds become so powerful that they cause physical pain (see table A.1 for examples). The precise level at which the pain sets in varies from person to person but is typically between 120 and 140 dB. Sounds above this threshold may still be audible but cause quasi-immediate (and possibly permanent) hearing damage. The risk of hearing damage depends on the level, frequency and duration of exposure, and the susceptibility of the individual [182: p. 204]. For long exposures – e.g. a few hours – the risk kicks in at about 85 dB (at 1 kHz).

A.3.3 Pitch

As indicated in figure A.3 different areas of the cochlea are responsible for sensing different frequencies. The lower or higher the frequency of a sound, the *lower* or *higher-pitched* it respectively appears to us. We should note that while the psychological and musical term *pitch* is closely related to frequency, both are not equivalent. Frequency is an objective, scientific measure, whereas pitch is a subjective measure part of our auditory sensation.

A.3.4 Loudness

Loudness is a psychological term for the attribute of auditory sensation in terms of which sounds can be ordered on a scale extending from "quiet" to "loud" [485]. In a sense, loudness is to sound pressure what pitch is to frequency: the former is a subjective measure while the latter is an objective, scientific measure to which the former is related. The correlation of loudness and sound pressure is complex. For one thing, besides sound pressure, loudness also depends on the frequencies present in a sound. In other words, our hearing is not equally sensitive (or *responsive*) to all frequencies.

A.3.4.1 Frequency-dependency

Pioneering research into loudness was conducted in 1930s by Fletcher & Munson (F&M). Most notably they studied the influence of sound pressure and frequency. Their main experiment can be summarised easily[16]. A group of listeners were exposed to a pure tone of 1 kHz at a particular SPL. Then, the tone's frequency was changed multiple times (between 20 Hz and 20 kHz) and each time all subjects were asked to indicate the SPL at which the new tone sounded equally loud as the original one. By averaging the answers a so-called *equal-loudness contour* was then defined, specific to the SPL at which the experiment started. This is a curve that connects points in the frequency-SPL plane – representing pure tones of a specific frequency played at a specific SPL – which humans perceive as equally loud. By repeating the experiment for other starting points, F&M established equal-loudness contours for SPLs of 0 to 120 dB in steps of 10 dB [181]. Later, many others have improved upon this work, eventually leading to the publication of a refined and extended set of contours in ISO standard 226 (last revised in 2003) [271]. Chart A.4 shows a subset of the standardised contours along with the corresponding originals defined by F&M, which differ quite significantly from the former.

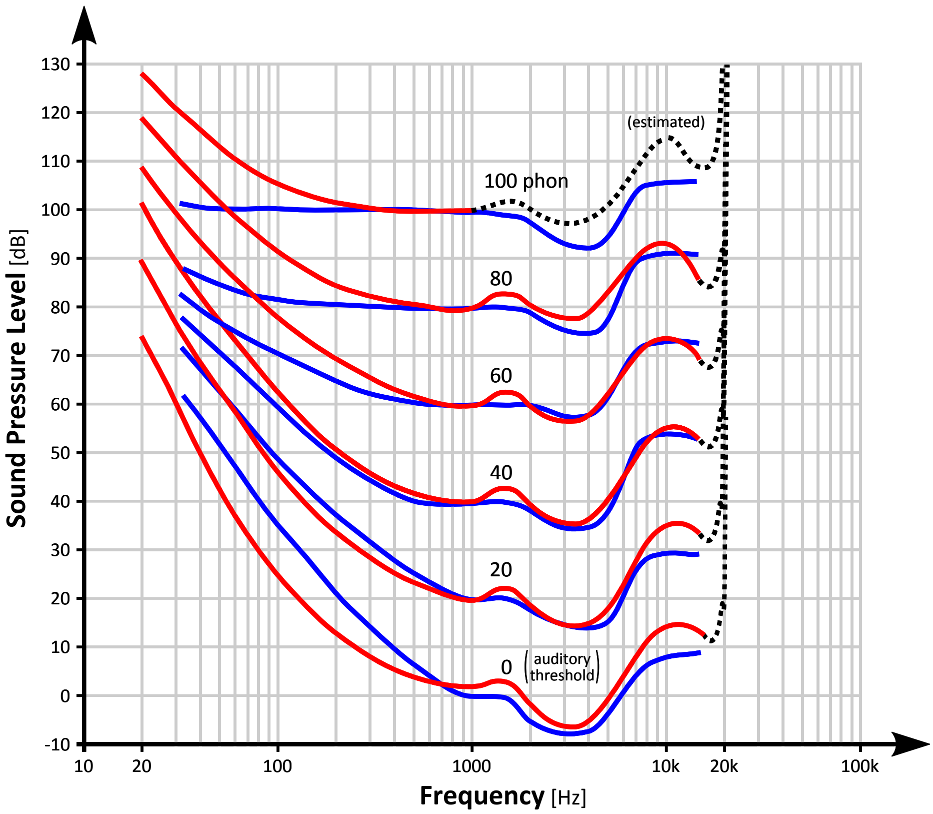


Chart A.4: Equal-loudness contours according to ISO 226:2003 [271] (in red) and the original Fletcher-Munson contours [181] (in blue)

The chart shows that loudness is indeed highly frequency-dependent. Consider the second ISO-contour from the bottom: even though there is considerable variation in physical strength (the SPL ranges from 15 to 90 dB) all tones on the curve sound as loud, or rather as quiet, as a 1 kHz tone at 20 dB. Because the curves become flatter at higher

[16] This description is a simplification (reproduced from [550]), see [181] for a full discussion of the protocol.

levels, we see that the frequency-dependency lessens with increasing sound pressure. Equal-loudness contours in fact describe the *frequency response* of our hearing, but that term is more commonly used in regard to man-made devices[17] than to biological systems.

A.3.4.2 Loudness measures

Each contour in chart A.4 is labelled with its *loudness level*, a measure denoted by L_N and expressed in *phon*. This unit was introduced by H. Barkhausen in 1925 [366]. By its current definition [269, 271] the loudness level of a sound is x phon when it is judged to be equally loud – by a person with normal hearing under specific listening conditions – as a 1 kHz tone at an SPL of x dB (i.e. the "starting points" in the described experiment). Equal-loudness contours, also known as *isophons*, are thus "lines of equal phons" and obviously the 0 phon-contour corresponds to the auditory threshold.

It is important to note that equal-loudness contours merely indicate an ordinal relation. For example, when a sound V has a loudness level of 70 phon and another sound W has a loudness level of 50 phon, a look at the chart tells us that V will be perceived louder than W, but not *how much* louder. It is not true, for instance, that sound V would be judged 40% louder (since $\frac{70}{50} = 1.4$), nor that it would be heard 10 times as loud (since at 1 kHz, $\frac{p_{\text{eff},V}}{p_{\text{eff},W}} = 10$). Instead, to most people, sound V will appear about 4 times as loud as sound W. Clearly then, there is no linear relation between (perceived) loudness and the loudness level in phon (or SPL in dB), nor between loudness and sound pressure in pascal. This has led to the introduction in 1936, by S. S. Stevens [502], of the *sone*, a unit which expresses *loudness*, denoted by N, on a scale proportional to our perception. By definition 1 sone is equivalent to the loudness of a 40 phon sound [269], for *loudness level* values $L_N \geqslant 40$ phon[18], the loudness N is calculated as:

$$N = 2^{\frac{L_N - 40}{10}} \quad \text{[sone]} \tag{A.17}$$

And consequently, for loudness values $N \geqslant 1$ sone[18], the loudness level[19] L_N is given by:

$$\Leftrightarrow \quad L_N = 40 + 10 \cdot \log_2 N \quad \text{[phon]} \tag{A.18}$$

This definition was chosen such that doubling a sound's loudness (in sone) effectively causes it to be perceived twice as loud by a person with normal hearing. When we apply equation A.17 to our example sounds V and W we find that they respectively have a loudness of 8 and 2 sone, thereby proving that V indeed sounds 4 times louder than W.

[17] E.g. a microphone or loudspeaker is said to have a "flat frequency response" when it is able to respectively capture or reproduce various frequencies present in a sound with (relatively) uniform accuracy.

[18] Formulas applicable for $L_N < 40$ phon and $N < 1$ sone can be found in [469].

[19] In a sense, loudness level (in phon) is to loudness (in sone) what sound pressure level (in dB) is to sound pressure (in Pa): the former expresses the latter on a logarithmic scale.

Table A.2 lists loudness levels between 40 and 140 phon with the corresponding loudness in sone.

Loudness Level (L_N)	Loudness (N)
140 phon	1024 sone
130 phon	512 sone
120 phon	256 sone
110 phon	128 sone
100 phon	64 sone
90 phon	32 sone
80 phon	16 sone
70 phon	8 sone
60 phon	4 sone
50 phon	2 sone
40 phon	1 sone

Table A.2: Loudness level values in phon and corresponding loudness values in sone

Equations A.17 and A.18 and the definition of the phon allow us to map loudness onto a physically measurable quantity – i.e. the SPL of a 1 kHz tone – and vice-versa. For example, 32 sone is the loudness of a 1 kHz tone at 90 dB. But what about pure tones at other frequencies? In that case we can rely on the equal-loudness contours. For a given pure tone sound, measuring the frequency and SPL allows us to look for the contour which passes through that point (or is the closest). This tells us the loudness level of the sound in question, which can then be converted into loudness by applying equation A.17. Conversely, for a given loudness, applying equation A.18 gives the loudness level, allowing us to pick the closest contour which connects all frequency-SPL pairs matching the loudness (level) in question. And what about sounds which, unlike pure tones, but like almost everything we hear in our daily lives, consist of multiple frequencies? To label such sounds in terms of loudness, we can experimentally look for the SPL at which a 1 kHz tone sounds as loud as the sound in question, which is inevitably a subjective judgment. When the SPL is x dB, the loudness level is x phon and applying equation A.17 gives the corresponding loudness.

Because loudness and loudness level are difficult to measure directly for anything but pure tones, these measures are not commonly used outside the lab. Instead, what is typically being measured in real-world contexts is SPL (as well as some derived measures which we will discuss in section A.5). However, for practical purposes it is of course convenient to have a general idea of how changes in sound pressure are perceived. Therefore table A.3 lists differences in sound pressure (level) with corresponding changes in perceived loudness which are (roughly) valid for much of the frequency range of our hearing. Useful things to remember are that the smallest perceivable SPL change is about 3 dB and that an increase or decrease of 10 dB respectively doubles or halves perceived loudness.

Change in Sound Pressure Level	Change in Sound Pressure (approximate)	Perceived loudness change
- 20 dB	÷ 10	¼ as loud
- 15 dB	÷ *5,62*	A lot quieter
- 10 dB	÷ *3,16*	Half as loud
- 5 dB	÷ *1,78*	Noticeably quieter
- 3 dB	÷ *1,41*	Slightly quieter (just perceptible)
+ 3 dB	× *1,41*	Slightly louder (just perceptible)
+ 5 dB	× *1,78*	Noticeably louder
+ 10 dB	× *3,16*	Twice as loud
+ 15 dB	× *5,62*	A lot louder
+ 20 dB	× 10	4 times as loud

Table A.3: Typical perception of sound pressure (level) changes

A.3.4.3 Other loudness-influencing factors

Although some aspects, like frequency-dependency, are relatively well understood, loudness is inherently a subjective concept – i.e. perception may vary from person to person – and remains the subject of active research[20]. It has been shown that, besides sound pressure and frequency, loudness perception is also affected by the duration of exposure as well as by various non-acoustical factors. The latter can be of a semantic, contextual or cognitive nature and are of particular importance when assessing the annoyance (or enjoyment) sounds may cause, which is often closely related to loudness itself. We will come back to this topic in our discussion of noise in chapter 4.

A.3.5 Psychoacoustics

The study and modelling of attributes of sound perception, such as loudness and pitch, is part of the wider field of psychoacoustics [178], the branch of science focused on the psychological and physiological responses associated with sound. The field has many practical applications in digital signal processing and audio compression (see section A.4), speech recognition and synthesis, etc.

A.4 Audio signals

The word "sound" is used well beyond its physical definition of audible vibrations of the air (or another medium). For example, gramophone records, audio cassettes, CDs or MP3 files are said to contain or carry sound[21] and telephone lines or radio waves are said to transmit it. Each of these 19th and 20th century inventions is founded on the same basic principle: the representation of sound as an *audio signal* (often simply called *audio*).

A.4.1 Analogue audio

An audio signal is an analogue representation of sound, usually as an electrical voltage. An *analogue* (or *analog*, in American English) *signal* is any continuous signal for which the time-varying feature is a representation of some other time-varying quantity, i.e. *analogous* to another time-varying signal. In the case of an audio signal the time-varying

[20] A recent and comprehensive overview of the state of the art in loudness research is provided by [182].

[21] In Dutch we call these *"geluidsdragers"*, which translates to "sound carriers", but English prefers the less evocative "sound recording media/formats".

feature is (usually) electrical voltage and the time-varying quantity being represented is sound pressure (see section A.2.2). Audio signals can be synthesized directly (i.e. with an analogue synthesizer) or may be generated by a *transducer*[22] such as a microphone or a musical instrument pickup (like on an electric guitar). A microphone is an acoustic-to-electric transducer: it converts actual (audible) sound into an electric signal. While there are many types of microphones they usually have a membrane or diaphragm which is set to vibrate by impacting sound waves, this in turn produces a wave of voltage changes corresponding to the air pressure fluctuations of the sound wave. Of course this correspondence is never prefect and microphones can have a different dynamic range[23], frequency response[23] and polar pattern (e.g. directional vs. omnidirectional).

Audio signals can be transmitted over electrical wires or radio waves, processed using analogue circuitry (e.g. amplifiers or equalizers), recorded on analogue storage media (e.g. audio cassettes or gramophone records) and/or fed back to loudspeakers[24] to reproduce the original sound (usually after being was amplified, such as in a concert setting).

To capture and reproduce the reverberation, ambience and spatial properties of live sound more accurately, it is common practice to use 2 or more independent audio signals in parallel. Each such *channel* is usually captured with a separate microphone and played back through a separate loudspeaker. Setups which involve only a single audio channel are said to produce *mono(phonic)* sound. Multi-channel setups are said to produce *stereo(phonic)* sound (from 2 channels) or *surround sound* (from more than 2 channels).

A.4.2 Digital audio

Nowadays virtually all transmission, processing and storage of audio happens by means of *digital* audio signals. These are sequences of discrete values, timed at specific (usually equal) intervals, which are electronically transmitted (i.e. in a binary representation).

Analogue audio signals can be converted into digital ones and vice-versa. The *digit(al)ising* of analogue signals (regardless of whether they convey audio) is performed by an electronic component called an *analogue-to-digital converter* (ADC or A/D). The reverse operation is done by means of a *digital-to-analogue converter* (DAC or D/A). Because there is no such thing as a digital microphone[25], all digital audio signals, except for synthetic ones (generated with a digital synthesizer), originate from digitalised analogue

[22] A transducer is a device that converts one form of energy into another.

[23] We explained this terms of human hearing in section A.3, the meaning in this context is analogous.

[24] A loudspeaker is an electric-to-acoustic transducer: it converts an electric signal into audible sound.

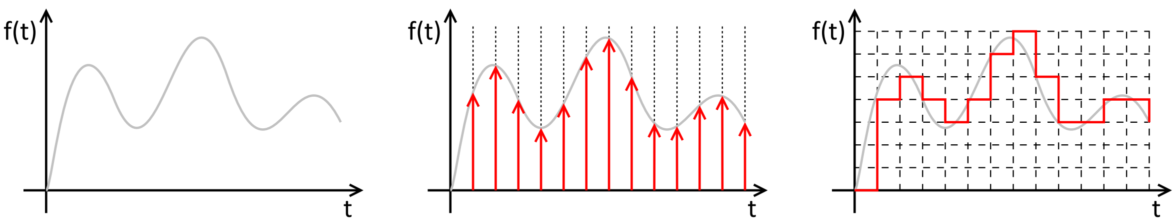
[25] In fact some companies do sell products labelled as "digital microphones" or "digital loudspeakers", but these devices usually contain a conventional analogue transducer that is coupled to some built-in ADC or DAC respectively [472].

audio signals. Similarly, because digital loudspeakers do not exist as such[25], digital audio signals must be converted into analogue ones before they can be played back.

Digital audio signals can be transmitted across digital circuits or radio waves, they can be processed using *digital signal processors* (DSPs) and/or recorded on digital storage media (e.g. CDs, DVDs or any type of computer memory). There are many different *formats* for the transmission or storage of digital audio, but before we discuss those we will take a more detailed look at the digitising of analogue (audio) signals.

A.4.2.1 Digitising

An analogue signal is a continuous-time signal which takes values from a continuous domain. A digital signal is a discrete-time signal which takes only discrete values. An ADC converts the former into the latter by means of two discretisation steps. First, the time domain is made discrete in a process called *sampling*. Here the ADC repeatedly measures (i.e. *samples*) the analogue input signal at discrete moments in time. Next, the value domain is discretised by *quantizing* each measurement or *sample*. *Quantization* is the process of approximating a continuous range of values (e.g. the voltage of an analogue audio signal) by a fixed, relatively small set of discrete values (e.g. the integers in [0, 7]). In other words, digital values outputted by an ADC are chosen from a predefined set. Figure A.4 illustrates both processes.



(a) Analogue signal: continuous time, continuous values

(b) Sampled signal: discrete time, continuous values

(c) Digital signal: discrete time, discrete values

Figure A.4: Analogue to digital signal conversion by means of sampling and quantization

The de facto way to represent an audio waveform digitally is *pulse-code modulation* (PCM). In this method the instantaneous amplitude of an analogue audio signal is sampled – typically by measuring the voltage – at uniform intervals, with each sample being quantized to the nearest value within a range of discrete steps. In figure A.5 we see how an audio signal is sampled and quantized to produce a 16-step PCM representation[26].

[26] In fact, this is an example of *linear* pulse-code modulation (LPCM) [588], a particular type of PCM in which quantization happens on a linear scale. This means sample values are proportional to the measured amplitudes. Other techniques may use a logarithmic or some other relation to map amplitudes onto sample values. Because LPCM is by far the most common form of PCM, the term PCM, though strictly more general, is often used instead of LPCM. For simplicity, we have henceforth use the term PCM to refer to what is effectively LPCM encoded audio – as we did in chapter 6 as well.

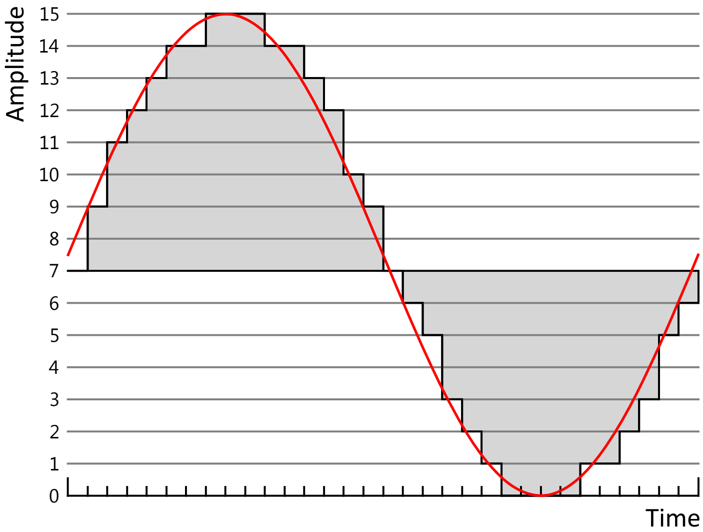


Figure A.5: The 4 bit pulse-code modulation (black/gray) of an audio signal (red)

Converting an analogue signal to a digital one (and back) typically results in some information loss. For audio this can cause a perceivable degradation of *sound quality*. Two basic properties determine the fidelity to the original signal: *sampling rate* and *bit depth*.

A.4.2.1.1 Sampling rate

Sampling rate, sample rate or sampling frequency defines the number of samples taken per unit of time. It is usually expressed in Hz or kHz. The higher the sampling rate, the more accurate the (voltage) fluctuations – and thereby the motion of the original sound wave – are captured and eventually reproduced upon playback. This is expressed formally by the *Nyquist–Shannon sampling theorem*, which states that a signal can be reconstructed perfectly (in theory) when the sampling rate is greater than twice the highest frequency present in the signal being sampled, or equivalently, when the highest frequency in the signal does not exceed the *Nyquist frequency* (half the sample rate).

By applying the theorem to the range of human hearing, which spans a bandwidth of about 20 kHz (see section A.3), we see that the minimum sampling rate required for perfect reconstruction of all audible sounds is around 40 kHz. For this and other technical reasons the sampling rate used on Audio CDs was chosen as 44.1 kHz. Other common sample rates for digital audio are 8, 11.025, 16, 22.050, 32, 48, 96 and 192 kHz.

A.4.2.1.2 Bit depth

Bit depth, resolution or sample size defines the number of bits used to store or transmit a single sample. This determines the number of possible digital values that each sample can take. In other words, the higher this number the bigger the set of values which can be chosen from during quantization. In the example in figure A.5 the bit depth is 4, which is not a common choice for audio signals. This results in 16 possibilities, shown as integers from [0, 15]. By contrast, the bit depth on Audio CDs is 16, allowing $2^{16} = 65536$ distinct voltage levels. Other common audio bit depths are 8, 20 and 24.

A.4.2.2 Digital audio formats

A digital audio format or (bit)stream format defines how one or more digital audio signals (i.e. channels), along with any auxiliary information and metadata, are represented as a single sequence of bits. Digital audio formats are often identified with an underlying physical (e.g. a CD) or logical (e.g. an MP3 file) storage medium. Nevertheless these are two distinct concepts: strictly speaking the format only applies to the data (i.e. the binary *content* of the CD or file), not the medium itself. An instance of a format (e.g. the contents of a particular CD or MP3 file) is called a *(bit)stream*.

Another distinction we should make is between *audio encoding* and *container* formats.

A.4.2.2.1 Audio encoding formats

An audio encoding format defines how the waveforms of one or more audio channels are represented as a bitstream. We already discussed one audio encoding format: PCM.

A device (or piece of software) responsible for producing streams in a particular encoding is called an *encoder*. In a sense an ADC that produces a PCM stream from one or more analogue audio signals is a PCM encoder. But usually an encoder takes an already digital (typically PCM-encoded) signal and converts it into another encoding format (e.g. MP3). A device (or piece of software) that converts an audio stream encoded in a particular format (e.g. MP3) to another format (typically PCM) is called a *decoder*[27]. Again, it could be argued that a DAC that converts a PCM stream into one or more analogue audio signals is in fact a PCM decoder.

Because streams produced by audio encoders only contain a description of the audio itself they are often called *"raw"* or *"headerless"* streams. Raw streams are commonly used for transmission within a single device but rarely as a means to transmit audio between devices or for persistent storage. The reason is that such streams typically cannot be correctly interpreted (i.e. *decoded*) without additional information about the properties of the stream. Examples are the number of channels in the stream, the sampling rate, the bit depth and last but not least the encoding format. One of the main purposes of

[27] A related term is *codec*, this portmanteau of (en)*co*der-*dec*oder (or *co*mpressor-*dec*ompressor), is used to refer to a device (or piece of software) that handles both (or either) the encoding and decoding of a digital data stream or signal. It is also commonly (but erroneously) used to refer to (en)coding *formats*.

container formats (see below) is precisely to incorporate such auxiliary information into the stream such that it becomes self-describing.

Encoding formats typically support various channel counts, sampling rates and bit depths (which is why these properties must be known to decode a raw stream). This allows trade-offs to be made between sound quality and storage space (or transmission bandwidth). Of course the total size of a stream also depends on the duration of the recorded sound. Therefore digital audio streams are typically described by their *bit rate*, which is the number of bits required per second of sound[28].

Many different audio encoding formats have been developed. The main distinction we should make is between *uncompressed* and *compressed* formats.

Uncompressed audio

In uncompressed audio encodings each sample of each channel is independently encoded using the exact amount of bits defined by the bit depth. This means that the bit rate of an uncompressed audio stream can typically be calculated by simply multiplying the number of channels, the sample rate and the bit depth. Consequently, changing one of these parameters has a predictable, linear effect: for example, choosing a sample rate of 16 instead of 32 kHz will half the bit rate and choosing a bit depth of 16 instead of 8, or using 2 channels instead of 1, will double it. The total size of a raw uncompressed stream can be calculated by multiplying its bit rate with its duration in seconds.

The (L)PCM format we discussed above is the de facto uncompressed audio encoding. It is the encoding used on Audio CDs, in which case there are 2 channels at a sample rate of 44.1 kHz and a bit depth of 16, resulting in a bit rate of 1411.2 kbit/s.

Compressed audio

Because high-quality uncompressed audio takes up considerable amounts of storage space or transmission bandwidth (e.g. 1 minute of CD-quality audio requires about 10 MB), many *compressed* audio encoding formats have been developed.

While lowering the channel count, sample rate or bit depth are effective, yet crude, ways of reducing the bit rate of digital audio, this is not how compressed audio encodings work[29]. Instead an encoder applies data compression algorithms to produce a stream with a bit

[28] Audio bit rates are usually expressed in kbit/s (= 1000 bits per second).

[29] Devices or software applications which change the channel count (though *up-* or *downmixing*), sample rate (through *resampling*) or bit depth (through *requantization*) of digital audio do exist but because their effect cannot be undone they are generally seen as DSPs rather than codecs.

rate that is lower[30] than the product of the channel count, sample rate and bit depth of the input stream. But when that compressed stream is fed to a corresponding decoder the latter will reproduce a stream with the same channel count, sample rate and bit depth as the stream that was fed to the encoder. However, the fact that these properties are preserved does not mean that the decoded stream is identical to the original. Whether it is depends on whether the compression scheme is *lossless*, as opposed to *lossy*.

Encodings which apply lossless compression allow an exact copy of the original bitstream to be reproduced upon decoding. As the name suggests, this means no sound quality is lost. The reduced bit rate is only "paid for" through the computational overhead caused by additional en- and decoding steps. Their lossless nature makes these audio encodings similar to general-purpose data compression formats (e.g. such as the ZIP file archiving format), but the former achieve better compression ratios[31] because the algorithms applied are specifically designed, or at least optimised, for the compression of digital audio. Some examples of lossless audio encodings are shown in table A.4.

Encodings which apply lossy compression effectively throw away information during the encoding process, such that the original stream cannot be restored exactly upon decoding. This means that, in exchange for the reduced bit rate, a permanent loss of sound quality is incurred, on top of a computation overhead. On the upside, this allows lossy encodings to achieve significantly better data compression ratios[31] than lossless ones. Moreover, lossy audio encoders use psychoacoustic techniques to filter out those parts of a signal which cause the least *perceivable* sound quality degradation possible within the space/bandwidth allotted by a target (average) bit rate. The best-known lossy audio encoding is MP3. Despite being surpassed in terms of sound quality (at the same or lower bit rates) by newer formats such as AAC, MP3 remains widely used. Table A.4 shows these and a few other examples of lossy audio encodings.

There is a wide choice of both lossy and lossless audio encodings out there. Generally they differ in terms of their intended use (e.g. general-purpose vs. specific types of audio[32]) and in the degree to which bit rate reduction is traded for computational overhead (and the latency related to that) and/or quality degradation. Most encoders can also be configured through numerous settings (e.g. to specify a target bit rate on lossy ones).

[30] Besides being decreased, the bit rate can also become variable when a technique called *variable bit rate* (VBR) encoding is used (as many audio compression formats do). This enables some intervals of the stream to be encoded with more or less bits than others, depending on their acoustic complexity. In such cases the encoded stream is better described by its *average* bit rate.

[31] The *compression ratio* achieved by a data compression algorithm on a particular piece of data is the ratio between the size of the compressed data and the size of the original uncompressed data. When we say a compression tool achieves a "better" compression rate than another this means the former is able to compress the same piece of data to a lower size than the latter can.

[32] For example, there is a whole class of lossy audio compression formats which are specifically designed for the transmission of voice signals in telephony systems.

		Encoding format	Contained by
Uncompressed		Linear pulse-code modulation (LPCM)	Audio CD (CDDA), DVD-Video, Blu-ray Disc WAVE file, AIFF file
Compressed	Lossy	Advanced Audio Coding (AAC)	MP4 file/stream
		Dolby Digital (A/52, AC-3)	DVD-Video, Blu-ray Disc
		DTS Coherent Acoustics	DVD-Video, Blu-ray Disc
		MPEG-1/2/2.5 Layer III (MP3)	MP3 file/stream
		Vorbis audio compression	Ogg file/stream
	Lossless	Apple Lossless Audio Codec (ALAC)	MP4 file/stream
		Dolby TrueHD	Blu-ray Disc
		DTS-HD Master Audio	Blu-ray Disc
		Free Lossless Audio Codec (FLAC)	Ogg file/stream

Table A.4: Examples of audio encoding formats and common associated container formats

A.4.2.2.2 Container formats

Container or wrapper formats serve to package or wrap raw streams in a single, self-describing *container stream*. We call a device (or piece of software) that wraps raw streams in a container stream a *wrapper* and one that interprets or unwraps a container stream a *parser*.

To make wrapped streams self-describing – to enable decoding without external information – parameters such as encoding format and settings, channel count and interleaving, sample rate, bit depth, byte order (*endianness*), *signedness*, etc., must be incorporated in the container stream itself (usually in a *header*). Containers often also add other auxiliary information such as error correction codes and synchronisation information[33]. Besides such technical information most container formats can also store human-readable metadata or *tags*. For example, MP3 files can be tagged with the name of the artist, the title of the song and album, the genre, etc. Obviously the addition of auxiliary information and metadata always causes a (usually minor) overhead in terms of storage space or transmission bandwidth compared to the raw stream(s).

Many container formats can accommodate multiple parallel streams in the same container stream though *multiplexing*. Besides audio, such formats typically handle video streams as well, and in some cases also other types of streams. For example, on a DVD or Blu-ray Disc one or more audio streams (for different languages), one or more video streams (different angles) and one or more subtitle streams are multiplexed in a single stream. Some container formats also support *chapters*, which are lateral subdivisions (e.g. tracks on an Audio CD, scenes on a DVD) that may cut across multiple parallel streams.

[33] Synchronisation information is needed to play back various multiplexed streams together (e.g. a video and an audio stream) and to enable instantaneous seeking to specific time points in the stream.

Table A.4 shows some examples of common container formats. While many containers can accommodate streams in different encodings some cannot. For example, CDDA, the format used on Audio CDs, only allows PCM-encoded audio streams with fixed parameters (see above). In some cases the distinction between the encoding format and the container format is not entirely clear. For example, MP3-encoded audio streams are typically stored in a dedicated container format only found in MP3 files.

Some containers are designed for persistent storage, on a physical or logical (i.e. a file) medium, while others are devised for transmission or *streaming* purposes. The formats used on Audio CDs, DVD-Video and Blu-ray Discs are examples of containers designed for persistent storage on physical media[34]. The WAVE [595] and AIFF formats, which are chiefly used to accommodate PCM streams, are examples of container formats intended for persistent, file-based storage. So are MP3 files, although techniques have been developed to stream MP3-encoded audio across the Internet. Container formats specifically designed for transmission (also called *transport stream formats*) are used in digital radio systems (e.g. DAB), digital television systems (e.g. DVB) as well as IP-based streaming technologies. Finally some container formats, most notably MP4 and Ogg, were designed with both persistent storage and streaming in mind.

A.4.3 Summary

Figure A.6 on the next page summarises our discussion of (digital) audio signals, by illustrating the recording and playback paths for (single-channel) digital audio.

At the top we see how sound is captured by a microphone producing an analogue audio signal, optionally processed and then digitalised by an ADC. Next, the uncompressed digital signal or stream may be digitally processed, encoded and/or wrapped and finally recorded on some medium by a recording device (or software). At the bottom we see how a digital audio stream is read from some medium by a playback device (or software) and is subsequently parsed (unwrapped), decoded and/or digitally processed. Next, a DAC converts the stream into an analogue audio signal, optionally processed and finally converted into audible sound by a loudspeaker.

As an alternative to persistent storage, a transmission or streaming scenario could be imagined by connecting both paths, by-passing the recorder and player (and thereby any storage medium). Both ends of the connected path could then reside on different (possibly remote) devices.

[34] Note that DVD-Video and Blu-ray Discs in fact use file-based container formats.

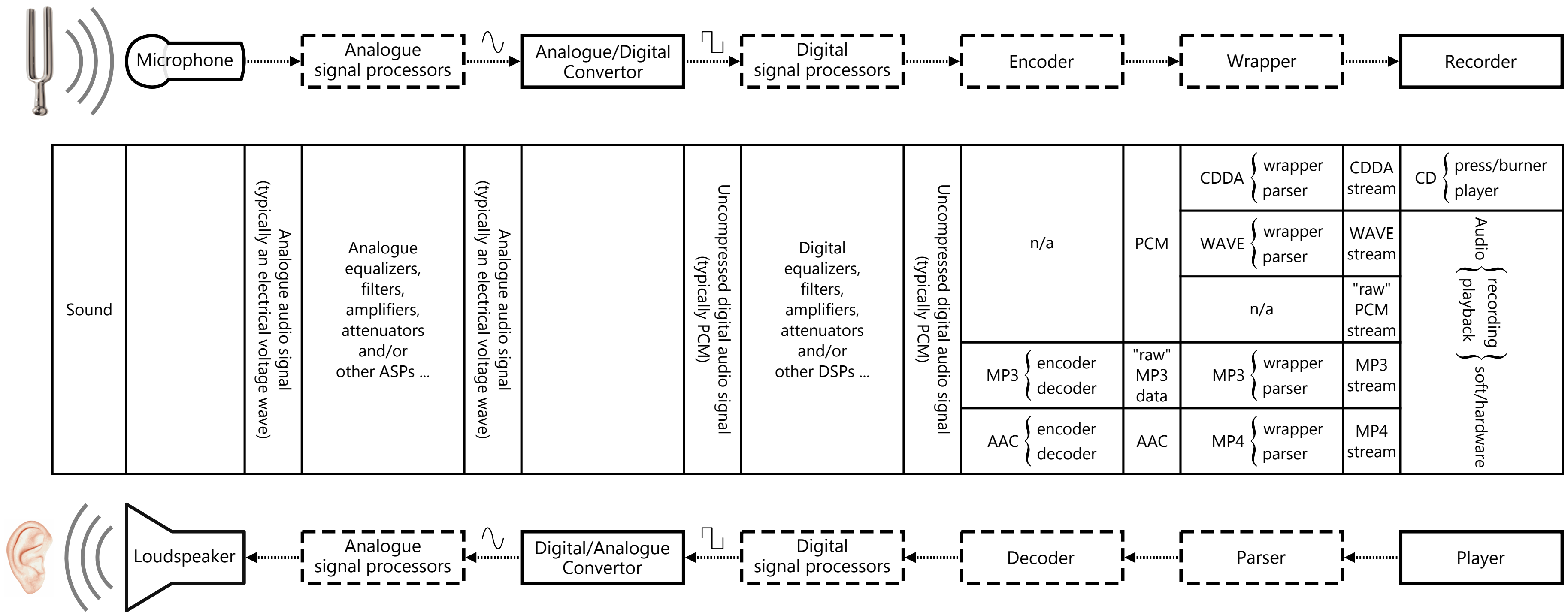


Figure A.6: Digital audio recording path (top) and playback path (bottom), with some clarifications and examples (middle). Boxes with a dashed outline represent optional stages.

A.5 Measuring sound

Now that we have reached a sufficient understanding of the physics of sound, its perception by humans and its representation by audio signals, it is time to discuss the devices which serve to measure physical properties of sound. We limit this discussion to two such devices, the *sound level meter* and the *dosemeter*. Because it is more relevant for this dissertation we will especially focus on the former type of device. The account presented here is by no means exhaustive, more details on these devices can be found in [50, 564].

A.5.1 Sound level meters

The *sound level meter* (SLM) is by far the most commonly used device to measure properties of sound. The basic purpose of this device is to measure sound pressure level – often shortened to sound level – which is a measure of the amplitude of sound waves. By definition, sound pressure level, abbreviated as SPL and symbolised as L_p, is twenty times the logarithm to the base ten of the ratio of the root-mean-square (RMS) of a given sound pressure to the reference sound pressure [268]. Which is precisely how we have defined it in equation A.5 on page 270. When it comes to measuring SPL, there are two important aspects that are not explicit in that definition. One is the way different frequencies in the sound are treated; the other is how the RMS value is obtained. Based on decisions made with regard to both aspects, different "SPL-like" quantities, called *sound level descriptors*, are measured, either using different kinds of SLM devices or different modes of a single device. Before explaining these decisions we examine a generic design which applies to virtually all current-day SLMs, as shown in figure A.7.

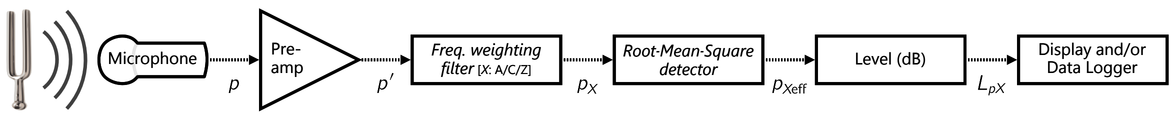


Figure A.7: Simplified design for a generic sound level meter

We see an end-to-end overview of components and connections found in a typical SLM. Predictably everything starts with a microphone[35], through which sound waves are converted into an analogue audio signal, and ends with a display, which shows measurements to the user. All components between these endpoints are essentially signal processors: they take an input, process it – thereby fulfilling a step in the calculation of the sound level descriptor – and output the result. Under each connection we have indicated which (intermediate) quantity is represented by the signal traveling across it. The audio signal

[35] Usually of a specialised kind, with a broad dynamic range and a (relatively) flat frequency range.

coming from the microphone, representing sound pressure (p), is fed to a *pre-amp(lifier)* in order to make it strong enough to be analysed in the subsequent steps. The conditioned signal (p') is then fed to a *frequency weighting filter*, of which there are different kinds (typically identified by a letter code). The choice for a particular one is the first of the decisions mentioned above. The signal, now representing frequency-weighted sound pressure (p_X, where X is the letter code of the weighting filter), is then fed to an *RMS detector* which computes a root-mean-square value of the input, which comes down to an averaging over time. There are multiple ways to do this and the choice for a particular RMS detector design, and configuration thereof, is the second of the decisions mentioned above. The signal, now representing the effective frequency-weighted sound pressure[36], is then fed to a component that computes *level* values in decibel, namely as 20 times the logarithm to the base ten of the ratio of the input to the reference sound pressure, as per the definition of SPL. Finally, these frequency-weighted SPL values (L_{pX}) are displayed. Apart from a display for direct readout of measurements, some SLMs – typically more expensive ones – also have a *data logging* feature allowing series of time-stamped measurements to be stored in memory for later analysis.

Historically, SLMs were built from analogue circuitry and had a mechanical display, usually a needle pointer. As we will see below, relics from that time are still present in some aspects of current SLM designs and practises. Today many, if not most, SLMs work with digital (i.e. sampled) audio signals, meaning that somewhere along the path from the microphone to the display sits an ADC (not shown in figure A.7) to convert the analogue signal into a digital one. All subsequent processors are then DSPs, implemented in hard- or software.

Figure A.8: A conventional sound level meter

Most sound level meters are easily portable and can be used either while being held in hand or mounted on a tripod. Figure A.8 shows a photo of a typical device. The black foam ball on top is a windscreen that is placed over the microphone to avoid interference from blowing wind while measuring outside. Usually it can be detached for indoor measuring.

As noted above the main decisions defining which sound level descriptor is being measured are the choice of a frequency weighting filter and the way the RMS detector averages the signal over time. In terms of the second criterion SLMs are split in two kinds: *conventional* and *integrating-averaging SLMs*. The latter kind is more advanced and thus usually more expensive than the former. Before

[36] This quantity is comparable to the *effective sound pressure* (ESP) as we defined it in section A.2.2, but is based on a frequency-weighted input and depending on the "second decision" the RMS detector may or may not work like equation A.4 (see section A.5.1.3).

we point out the difference between both kinds, we first discuss the standardisation and accuracy of SLMs, and explain what frequency weighting filters are.

A.5.1.1 Standards & accuracy

Sound level meters, sound level descriptors, and SLM operating procedures are the subject of various standards and norms, issued by national and international bodies. For our purposes the most relevant current standards, issued by the American National Standards Institute (ANSI) and the International Electrotechnical Commission (IEC), are: ANSI S1.4-1983 [15] for conventional SLMs, ANSI S1.43-1997 [17] for integrating-averaging SLMs and IEC 61672-1:2002 [268] for both kinds[37]. The corresponding ANSI and IEC standards are generally compatible, but details may differ.

For measurements to be accurate manufacturers must carefully determine the precise characteristics of all SLM components, which are inextricable linked to one another[38]. Moreover, SLMs have to be *calibrated*. As explained in appendix B, calibration is a process in which systematic measuring errors are detected, by comparison with a trusted reference device, and are then corrected for. In order to meet predefined error tolerance limits SLMs are usually calibrated in an acoustic lab by the manufacturer or reseller before it is sold. Devices that underwent such a procedure usually come with a document that proves their accuracy. Such *certificates* are typically sanctioned by an official entity and describe the calibration procedure – for which there are norms as well – and list the results. The inclusion of a certificate is often a determining factor in the retail price of an SLM. Because various factors[39] can cause slight changes in accuracy over time, some SLMs – again, typically more expensive ones – also come with a matching *calibrator*. This is a separate accessory which produces sounds with precisely known characteristics[40], that when slid over the SLM's microphone allows the meter to recalibrate itself.

SLMs are divided into categories based on their accuracy. The ANSI standards [15, 17], and the old IEC standards [265, 266], specify four categories, *Type 0*, *1*, *2* and *3*, with progressively looser tolerances, respectively ±0.4, ±0.7, ±1.0 and ±1.5 dB. The current international standard, IEC 61672-1:2002 [268], retains just two categories, *Class 1* and *2*, corresponding roughly to Type 1 and 2 respectively. In this standard tolerances are specified per frequency and are generally broader compared to Type 1/2 because they now also include allowances for uncertainties of measurement [268: p. 12].

[37] IEC 61672-1:2002 has replaced and unified two earlier standards, IEC 60651:2001 [265] for conventional SLMs, and IEC 60804:2000 [266] for integrating-averaging SLMs. IEC 61672-1:2002 has also been adopted as *European Norm* EN 61672-1:2003.

[38] For instance, the sensitivity and frequency response of the microphone and the *gain* applied by the pre-amp cannot be chosen independently.

[39] E.g. mechanical shocks and humidity or temperature changes.

[40] Typically a 1 kHz pure tone at 94 or 114 dB.

In terms of intended purposes the different categories can be described as follows:

- Type 0 units are meant as a reference for the calibration of other SLMs in labs;
- Class/Type 1 units are high-precision SLMs;
- Class/Type 2 units are general purpose SLMs;
- Type 3 units were only meant for surveys with lower accuracy demands.

Official regulations for sound measurements typically demand that an SLM of class/type 2 or better is used.

	Conventional SLM	Integrating-averaging SLM
Class 1	€ 1500–2250	€ 1680–6240
Class 2	€ 280–1450	€ 1030–5550
Not compliant	€ 180–340	n/a

Table A.5: Retail prices ranges of IEC 61672-1 class 1 and 2 compliant and incompliant sound level meters (source: noisemeters.co.uk)

Because it is generally easier to construct a less accurate instrument, the type or class of a device is usually reflected in its price. Table A.5 gives a general idea of retail prices of IEC 61672-1 compliant SLMs as well as devices which do not meet current standards. The price of an SLM also depends on whether it is a conventional or an integrating-averaging device, which additional features (e.g. data logging) it has, which certificates and accessories come with it, and so forth.

A.5.1.2 Frequency Weighting

In an effort to let SPL measurements correspond better to subjectively perceived loudness – which is frequency-dependent (see section A.3.4.1) – SLMs typically apply a *frequency weighting filter*. In general, such a filter analyses the frequencies present in a signal and constantly adjusts the signal by applying a relative weight to each frequency. In case of acoustical frequency weighting filters, as found in SLMs, this usually means that frequencies where our hearing is most sensitive (typically 3-6 kHz) are emphasised and those where it is less sensitive (typically very low and very high frequencies) are attenuated. For example, if a sound X is comprised of frequencies between 3 and 4 kHz and a sound Y contains frequencies between 50 and 200 Hz, and both sounds are physically equally powerful, then an SLM with a filter would rate sound X stronger (i.e. *louder*) than sound Y, whereas an SLM without one would rate them equally.

Various acoustical frequency weighting filters have been defined, based on different models of the frequency response of human hearing and/or aimed at different application domains (e.g. general purpose, concert settings, industrial settings, etc.). Some examples are the *A-weighting* filter – by far the most commonly used one – and the *B*-, *C*-, *D*- and *Z-weighting* filters. Chart A.5 shows the *weighting curves* which define these filters. For pure tones between 10 Hz and 20 kHz, each curve describes the weighting, as a dB offset[41] to be added to the unweighted SPL[42], which is applied by the filter in question.

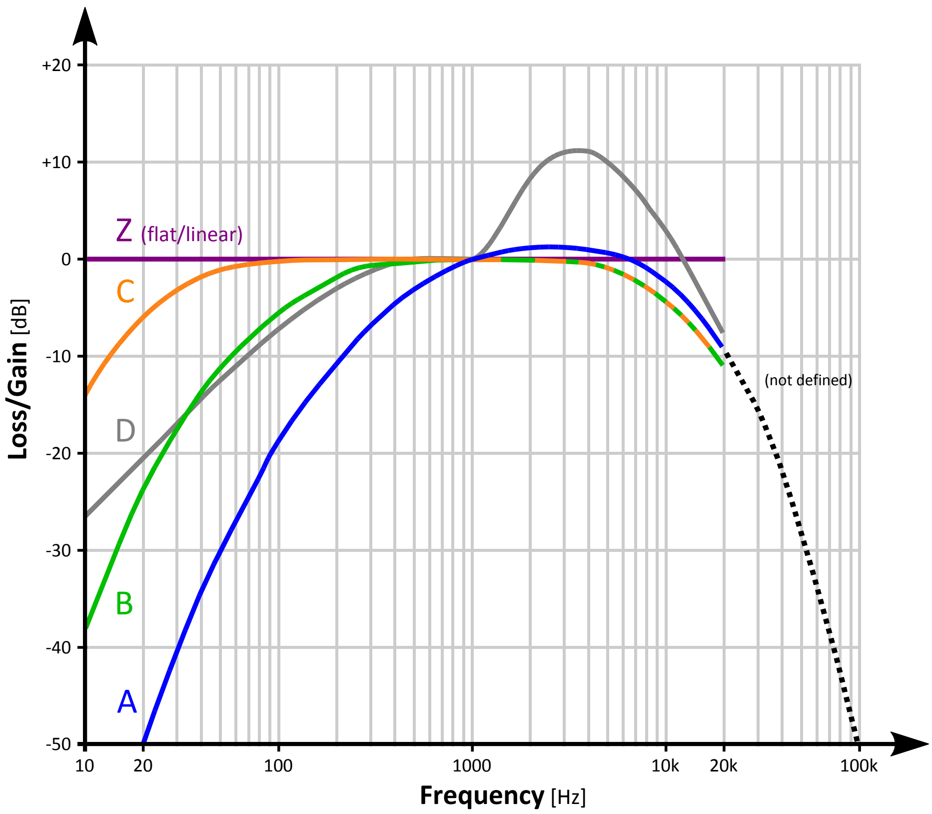


Chart A.5: Acoustic weighting curves A (blue), B (green), C (orange), D (gray) and Z (purple)

Typically the use of a frequency weighting filter is indicated by expressing measurements in dB(X), instead of dB, where X is the letter code of the filter. For example, as we can see in the chart, a 200 Hz tone with an SPL of 60 dB would be measured as 49.1 dB(A) by an SLM which applies A-weighting (as the A-weighting offset for 200 Hz is -10.9 dB). As with research into the frequency-dependency of loudness, the frequency of 1 kHz was taken as a "neutral" reference point at which the filters do not alter the signal. This means that for pure tones at 1 kHz the offset is always 0 dB, such that: y dB $\Leftrightarrow$ y dB(X).

[41] When the offset is positive – meaning the frequency is amplified – we call it a *gain* and when it is negative – frequency is dampened – we call it a *loss*.

[42] Note that while we are describing weighting curves by means of dB offsets (as do the ANSI and IEC standards), in actual SLMs frequency weighting is applied *before* the RMS and the level are computed, as illustrated by figure A.7, because after these operations all information about frequencies is lost.

The A- and B-weighting curves are based on the work of Fletcher and Munson [181], and were first published in 1936 by the Acoustical Society of America (ASA) as part of a tentative American SLM standard adopted by the ANSI [4]. The A-weighting curve is based on the 40 phon F&M contour (see chart A.4). Basically the weighting curve is an approximation of the inverted, normalised (i.e. shifted to 0 dB at 1 kHz) equal loudness contour. Strictly speaking it is thus only valid for relatively quiet sounds[43]. The ASA recognised this and recommended that the flatter B-weighting curve, which is based on the 70 phon F&M contour, would be used for louder sounds. Later ANSI standard revisions, up to the current one [15], added the even flatter C-weighting curve, that is also intended for loud sounds (e.g. > 100 dB) and peak measurements (see section A.5.1.5). The ANSI weighing curves were subsequently adopted by international standardisation bodies such as ISO and IEC. In 1976 the IEC introduced the D-weighting curve devised for assessing loud aircraft pass-over noise [264], but that standard has since been withdrawn. B-weighting has fallen into disuse and while the inclusion of a B-weighting filter is still mandatory for full compliance with the American SLM standard [15] it is no longer required by its international counterpart [268].

The use of A-weighted measurements was cemented into place when, in the late 1960s, various regulatory agencies began mandating it for the assessment of noise exposure [504]. Today, even though it was never intended to be applied so broadly, and despite being based on outdated research[44] and the existence of ample scientific evidence of its unsuitability[45], A-weighting is used commonly, and often mandatory, for the measurement of environmental [272, 273] and occupational noise [365: p. 33] (whether loud or not) and for the assessment of hearing damage risk. As discussed in chapter 4 it is imposed either directly, because laws or regulations explicitly require agencies to collect A-weighted measurements in various contexts, or indirectly, because legal (or other) limits on noise exposure in various contexts are typically expressed in dB(A), necessitating A-weighted measurements to be made in order to enforce them (i.e. when looking for violations).

The current international standard on SLMs [268] requires that users can choose between three weighting modes: A-, C-, and the "new" *ZERO* or Z-weighting. For selected frequencies, from 10 Hz to 20 kHz, a table lists the weighting offset to be applied in each mode along with an error tolerance limit per SLM class[46]. As the name suggests, the Z(ERO) weighting mode in fact does not apply a weighting at all: the offset is 0 dB

[43] And in fact only for pure tones.

[44] While loudness research after the 1930s has led to the establishment of improved, standardised equal loudness contours [271], the definitions of the A- and B-weighting curves have never been revised accordingly and remain based on the original F&M contours.

[45] Ironically multiple studies have shown that the now obscure B-weighting scale correlates better with subjective loudness perception than A, C and D-weighting [504].

[46] For example, the A-, C- and Z-weighting of a 40 Hz tone must respectively result in offsets of -34.6, -2.0 and 0 dB, with a tolerance of ±1.5 dB on a class 1, and ±2.5 dB on a class 2 device; for a 1 kHz tone all offsets are 0 dB, with tolerances of ±1.1 and ±1.4 dB; and for a 2.5 kHz tone the offsets are +1.3, -0.3 and 0 dB, with tolerances of ±1.6 and ±3.1 dB [268: p. 16].

across the frequency range, as illustrated by the completely flat purple line in chart A.5. The IEC added Z-weighting to the standard to replace the "flat", "unweighted" or "linear" mode, which virtually all SLMs were already fitted with, because the actual frequency range over which the response was linear[47] varied from one manufacturer to another.

Besides defining weightings and tolerance limits for specific frequencies, the SLM standards [15, 268] also describe the weighting filters as mathematical functions. For a frequency $f \in [10\,\text{Hz}, 20\,\text{kHz}]$, the A-, C and Z-weighting, again as a dB offset, is respectively calculated from equation A.19, A.20 and A.21:

$$A(f) = 20 \cdot \log_{10}\left[\frac{f^4 \cdot {f_4}^2}{\left(f^2 + {f_1}^2\right) \cdot \sqrt{\left(f^2 + {f_2}^2\right) \cdot \left(f^2 + {f_3}^2\right)} \cdot \left(f^2 + {f_4}^2\right)}\right] + A_{1000} \quad [\text{dB}] \tag{A.19}$$

$$C(f) = 20 \cdot \log_{10}\left[\frac{f^2 \cdot {f_4}^2}{\left(f^2 + {f_1}^2\right) \cdot \left(f^2 + {f_4}^2\right)}\right] + C_{1000} \quad [\text{dB}] \tag{A.20}$$

$$Z(f) = 0 \quad [\text{dB}] \tag{A.21}$$

Where the frequencies f_1, f_2, f_3 and f_4 have the following values:

$$\begin{aligned} f_1 &= 20.598997 \text{ Hz} \\ f_2 &= 107.65265 \text{ Hz} \\ f_3 &= 737.86223 \text{ Hz} \\ f_4 &= 12194.217 \text{ Hz} \end{aligned}$$

And in which A- and C-weightings of 0 dB at 1 kHz are ensured by the *normalisation constants* A_{1000} and C_{1000}, with values:

$$\begin{aligned} A_{1000} &= 1.9997 \text{ dB} \\ C_{1000} &= 0.0619 \text{ dB} \end{aligned}$$

As the first acoustic frequency weighting curves were defined during the early days of electronics, the filters, and SLMs as a whole, were built from analogue circuitry. Today's digital SLMs apply frequency weightings by means of a DSP. Generally speaking this can work in two ways. One is to convert the signal from the time domain to the frequency domain – usually through a *Fourier transform* – after which individual frequencies are weighted using equations A.19 to A.21. The other is to process the time domain signal with a digital filter [262: pp. 317–341, 414: pp. 200–275] which approximates the desired weighting curve, as we did in NoiseTube Mobile (see section 6.5.4.2).

[47] Remember that, just like human hearing, microphones are characterised by a frequency response curve which is not necessarily flat.

Even though there may be better alternatives (also beyond the curves discussed here) to A-weighting in many of its current use cases, for all the reasons given in section A.3.4 it remains inherently difficult to make accurate estimates of perceived loudness from weighted SPL measurements alone. This is one of the reasons why SLMs aimed at professional acousticians often provide additional frequency spectrum analysis features which may help to estimate loudness[48]. But such devices tend to be prohibitively expensive and difficult to use for people lacking appropriate training. When weighted SLP measurements are all one has, interpretation can be facilitated through a scale, established for the weighting in question, which associates particular levels with examples from daily situations (like we did for unweighted SPLs in table A.1) and by keeping in mind how additive changes in SPL tend to be perceived (see table A.3).

A.5.1.3 Averaging sound pressure over time

As we explained in section A.2.2, instantaneous sound pressure is neither a practical nor a very interesting way to quantify the strength of sound. To measure various sound level descriptors[49] an SLM must therefore somehow *average* the time-varying sound pressure – represented by the signal coming from the microphone – before the *level* can be computed. Yet, as we explained, the arithmetic mean of the sound pressure is not a suitable average because it is meaningless and possibly zero. Instead, SPL is by definition based on the *effective sound pressure*, which is computed as the RMS of the time-varying sound pressure over a time interval.

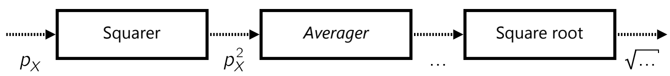


Figure A.9: Detail of the *Root-Mean-Square detector* from figure A.7

In an actual SLM, the RMS value is determined, or "detected", by an RMS detector, the basic layout of which is shown in figure A.9. First the frequency-weighted sound pressure is squared, then somehow averaged, and finally the square root of the result is returned. The way the "averager" works depends on the kind of SLM.

In conventional SLMs, also called *exponentially averaging SLMs*, an *exponential average*, characterised by a *time constant*, is computed. The square root of this exponential average is a "*quasi*-RMS" value, an approximation of the true RMS [552]. Computing the level from the quasi-RMS results in a *time-weighted sound level* descriptor (see below).

[48] A better solution still would be to use a *sonemeter* or *loudness meter* [182: pp. 215–216], instead of an SLM. Such a device analyses the frequency spectrum in a way that goes beyond a simple frequency weighting filter, to show loudness values in sone, which more closely correspond to perceived loudness for most people. However, because sonemeters are a recent innovation they are currently not commonly used and may be difficult, if not impossible, to find commercially.

[49] With the notable exception of *peak sound level*, which we will discuss in section A.5.1.5.

In integrating-averaging SLMs, also called *linear integrating SLMs*, a *linear average* over a specific *averaging time* is computed. The square root of this linear average is a true RMS value. Computing the level from that then results in a *time-average sound level* descriptor, better known as *equivalent continuous sound level* (also discussed below).

The reason for the existence of both averaging methods is historical. Traditionally, exponential averaging (also called *time weighting*) was used because at first it was not technically feasible to compute linear averages [50]. Today, linear averaging is typically preferred [90] but the conventional method remains widely used as well. This is due to various regulations which still demand time-weighted sound level measurements to be made, and to the higher cost of integrating-averaging SLMs compared to conventional ones. Integrating-averaging SLMs are mainly marketed to professional acousticians and usually also include the functionality of conventional SLMs – i.e. they can measure time-weighted as well as time-average sound level, sometimes even simultaneously.

A.5.1.3.1 Time-weighted sound level

In the early days of analogue SLMs it was not possible to compute the true RMS of the time-varying sound pressure. Because driving the needle pointer directly with the "levelled" squared time-varying sound pressure would have resulted in an unreadable blur, an electrical damping resistor was used to literally "slow down the meter" such that it became readable [50]. This resulted in an RMS detector circuit that computed a continuous *moving average* which asymptotically rose (or fell) to the true RMS with some delay. When measuring *time-weighted sound level*, which is formally defined below, today's SLMs still use detectors that apply the same principle[50], called *exponential averaging* or also *(exponential) time-weighting*[51], to approximate the true RMS. Equation A.22 gives a mathematical definition for this "quasi-RMS", at some observation time t:

$$p_{\mathrm{qrms}}(t) = \underbrace{\sqrt{\underbrace{\frac{1}{\tau}\underbrace{\int_{-\infty}^{t} \underbrace{\underbrace{p_X^2(\xi)}_{\text{Squaring}} \cdot e^{\frac{-(t-\xi)}{\tau}}}_{\substack{\text{Exponential}\\ \text{time-weighting}}} \,\mathrm{d}\xi}_{\text{Exponential integration}}}_{\substack{\text{Exponential averaging}\\ =\text{"Quasi mean-square"}}}}}_{=\text{"Quasi root-mean-square"}} \tag{A.22}$$

The squared instantaneous sound pressure is *time-weighted* through multiplication with an exponential term[52]. This causes recent (w.r.t. t) sound pressure values to contribute

[50] Yet, thanks to advances in technology, it may be implemented differently.

[51] Looking at equation A.22 we see that *(exponential) time-weighting* is in fact a pars pro toto name.

[52] This is why we talk about *exponential* averaging and *exponential* integration.

more "heavily" to the average than earlier ones. The *response time* of the detector is only determined by the *(exponential) time constant*, τ. Concretely, τ represents the time (in seconds) it takes for the quasi-RMS to reach $\sqrt{1-\frac{1}{e}} \approx 79.5\%$ of its final asymptotic value, i.e. the "real" RMS.

On old SLMs, selecting the desired time weighting configuration meant switching between different resistor circuits, each characterised by a specific time constant. Since those days there have been three popular time weightings, originally called *Slow*, *Fast* and *Impulse* and now usually identified simply by the letter codes *S*, *F* and *I*. Table A.6 lists their defining time constants. In case of time weighting I a different time constant is used when the squared instantaneous sound pressure is rising than when it is decaying. Today's SLMs typically still allow the user to choose between different time weightings. The current international SLM standard, IEC 61672-1:2002 [268], no longer mentions time weighting I and only requires time weightings S and F to be supported.

		Time constant (τ)	
		Rise time	Decay time
Time weighting	S (Slow)	1.0 s	
	F (Fast)	0.125 s	
	I (Impulse)	0.035 s	1.5 s

Table A.6: Time constants for time weightings

When time weighting is used the measured quantity describes *time-weighted sound level*, which is formally defined as: twenty times the logarithm to the base ten of the ratio of a given root-mean-square sound pressure to the reference sound pressure, the root-mean-square sound pressure being obtained with a standard frequency weighting and standard time weighting [268]. In symbols, the τ-time-weighted, X-frequency-weighted sound level at any instant of time t, is given by:

$$L_{X\tau}(t) = 20 \cdot \log_{10}\left[\frac{\sqrt{\frac{1}{\tau}\int_{-\infty}^{t} p_X^2(\xi) \cdot e^{\frac{-(t-\xi)}{\tau}}\, d\xi}}{p_0}\right] \quad [\text{dB}(X)] \qquad (A.23)$$

In which:

- X is the letter code of the chosen frequency weighting filter, which according to IEC 61672-1:2002 can be A, C or Z;
- τ in the LHS is the letter code of the chosen time weighting, which according to IEC 61672-1:2002 can be F or S, and τ in the RHS is the corresponding time constant in seconds;
- ξ is a dummy variable of time integration from some time in the past, indicated by -∞, to the time of observation t;
- $p_X(\xi)$ is the X-frequency-weighted instantaneous sound pressure at instant ξ;
- p_0 is the reference sound pressure, taken as 20 µPa.

The numerator of the argument of the logarithm in equation A.23 is the quasi-RMS we saw in equation A.22. More formally defined, it is the exponential-time-weighted, root-mean-square, frequency-weighted sound pressure at observation time t.

In terms of sound levels, the meaning of the time constant τ can be understood as follows: when the "real" SPL of a sound is instantaneously raised to y dB, then the measured time-weighted sound level will reach $\approx y - 2$ dB within one time constant[53]. After that, if the SPL is kept at y dB, the time-weighted sound level will continue to asymptotically approach y dB. The choice of a particular time weighting can thus significantly affect the results one gets, especially when the SPL fluctuates quickly.

Depending on the chosen frequency weighting X and time weighting τ, different time-weighted sound level descriptors, denoted $L_{X\tau}$, can be measured. An IEC 61672-1:2002-compliant SLM should support at least six of those: L_{AS}, L_{AF}, L_{CS}, L_{CF}, L_{ZS} and L_{ZF}. Because each measurement represents a "snapshot" of the current SPL, time-weighted sound level is sometimes called *instantaneous sound pressure level*. Moreover, because it is the traditional way to measure SPL, it is also referred to as *conventional sound level* or simply *sound level*, and denoted by L_X, L_{pX} or just L_p. Hence, whenever it is not clearly indicated one should always check which frequency- and time weighting are being used.

In an analogue SLM the time-weighted sound level is a continuous variable, which can be used to drive an analogue display such as a needle pointer. In a device with a digital display or logger, sequences of discrete $L_{X\tau}(t)$ values must be computed or sampled. The number of values which are generated per unit of time is not defined by the SLM standards and is independent of the time constant τ. When values are only shown on a numerical display, the rate is usually about 1 Hz (more would hinder readability), but when there is a graphical display or the data is being logged, it can be as high as 10 Hz.

In a device working with digital audio signals, each $L_{X\tau}(t)$ value can be calculated from discrete frequency-weighted samples, leading up to t, by approximating the integral in equation A.23 with a Riemann sum, in which case the product of the chosen time constant (in seconds) and the sampling rate (in hertz) must be taken as the value for τ in the RHS of the equation. Alternatively, time-weighting of digital audio signals can also be implemented as an autoregressive–moving-average (ARMA) filter, such as in [354].

A.5.1.3.2 Time-average sound level

Because they can faithfully follow sound pressure fluctuations, integrating-averaging SLMs allow to measure SPL based on true RMS values calculated over intervals of a specific, finite duration. Their RMS detector integrates the squared instantaneous sound

[53] Because $10 \cdot \log_{10}\left(1 - \frac{1}{e}\right) \approx 2$.

pressure over a fixed interval and divides the result by the interval's duration to obtain a *linear* (i.e. non-time-weighted) *average*. Hence the name *integrating-averaging* SLM.

With such a device one can measure *time-average sound level*, better known as *equivalent continuous sound level* or simply "the L_{eq}", which is defined as: twenty times the logarithm to the base ten of the ratio of a root-mean-square sound pressure during a stated time interval to the reference sound pressure, sound pressure being obtained with a standard frequency weighting [268]. In symbols, the time-average, X-frequency-weighted sound level over an interval with duration T is given by:

$$L_{XT} = L_{X\mathrm{eq}T} = 20 \cdot \log_{10} \left[\frac{\sqrt{\frac{1}{T} \int_{t-T}^{t} p_X^2(\xi)\, \mathrm{d}\xi}}{p_0} \right] \quad [\mathrm{dB}(X)] \tag{A.24}$$

In which:

- X is the letter code of the chosen frequency weighting filter, which according to IEC 61672-1:2002 can be A, C or Z;
- ξ is a dummy variable of time integration over the interval ending at the time of observation t;
- T is the averaging time, the duration of the averaging interval;
- $p_X(\xi)$ is the X-frequency-weighted instantaneous sound pressure at instant ξ;
- p_0 is the reference sound pressure, taken as 20 µPa.

The numerator of the argument of the logarithm in equation A.24 is the "true" root-mean-square, frequency-weighted sound pressure over an interval of duration T. When Z-frequency weighting (i.e. no actual frequency weighting) is used this quantity is identical to what we defined as *effective sound pressure* in section A.2.2 (for an equal interval). Because no time-weighting is involved, all sound pressure values since the start of the interval are included with equal importance in the computation of a time-average sound level value – i.e. the averaging and integration are indeed *linear*, as opposed to exponential.

Depending on the chosen frequency weighting X and the averaging time T, different time-average sound level descriptors, denoted L_{XT} or $L_{X\mathrm{eq}T}$, can be measured. For example, the A-weighted time-average sound level over 1 hour would be denoted as L_{A1h} or L_{Aeq1h}[54]. Sometimes only the shorthand L_{eq} is mentioned, in which case one should always check which frequency-weighting was used and over which interval measurements were made.

Since there is no time constant (τ) involved, by definition there is no such thing as "slow L_{eq}" or "fast L_{eq}". Still, some SLMs (can) compute time-average sound level values with exponential integration/averaging, or by averaging of sampled time-weighted sound

[54] Sometimes commas or spaces are added to improve readability, e.g. $L_{\mathrm{Aeq,1s}}$ or $L_{\mathrm{Aeq\ 1s}}$.

level values. While this does not necessarily lead to significant errors, especially not if F(ast)-time-weighting is used, it is not correct with respect to IEC61672-1:2002 [268] and should not be used unless regulations explicitly call for it.

Because each value is computed over a separate time interval, time-average sound level is by definition a discrete variable. Most current integrating-averaging SLMs internally work with digital audio signals. This means the device contains an ADC that samples the instantaneous sound pressure – represented by the analogue signal coming from the microphone – thousands of times per second, such that the digital signal closely follows its fluctuations. Each $L_{\mathrm{Xeq}T}$ value is then computed by approximating the integral in equation A.24 with a Riemann sum over a series of $T \cdot f_s$ squared, frequency-weighted samples, where T is the interval duration in seconds and f_s is the sampling rate in hertz. This gives:

$$L_{\mathrm{Xeq}T} = 20 \cdot \log_{10}\left[\frac{\sqrt{\frac{1}{(T \cdot f_s)}\sum_{i=1}^{(T \cdot f_s)} p_X^2(i)}}{p_0}\right] \quad [\mathrm{dB}(X)] \qquad (\mathrm{A.25})$$

As noted, time-average sound level is commonly called *equivalent continuous* sound level, which derives from the fact that it equals the constant SPL at which an imaginary sound would produce the same total sound energy as the actual measured sound (with fluctuating SPL) during the averaging interval. This is illustrated by chart A.6, in which the continuous (as in non-discrete), F-time-weighted sound level curve (L_{ZF}) can be interpreted as an approximation of the fluctuating SPL of the actual sound.

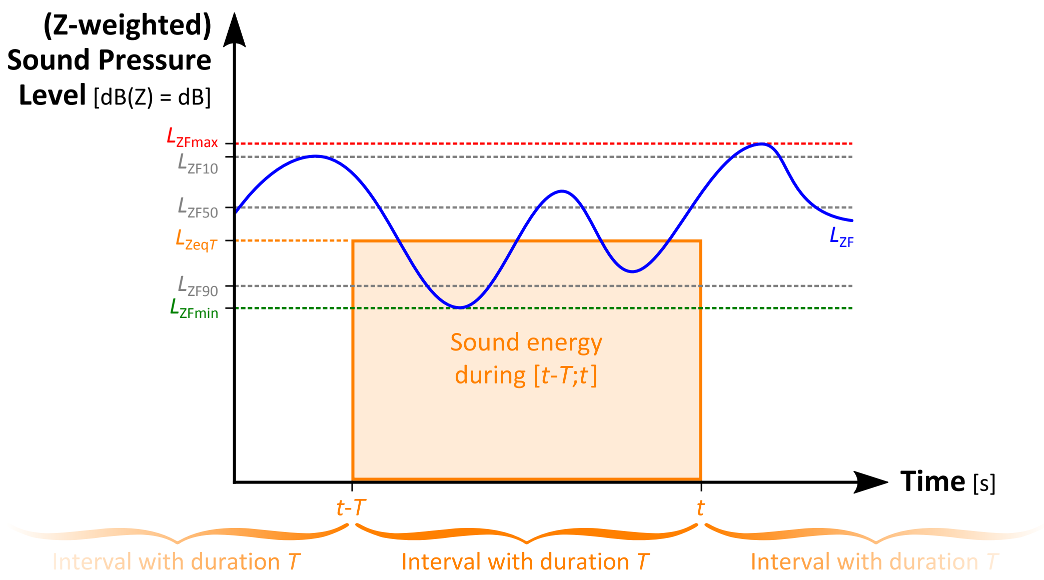


Chart A.6: Different (Z-weighted) sound level descriptors: F-time-weighted sound level (L_{ZF}), time-average sound level over one interval of T long ($L_{\mathrm{Zeq}T}$), maximum (L_{ZFmax}) and minimum (L_{ZFmin}) sound level and percentile sound levels (L_{ZF90}, L_{ZF50} and L_{ZF10})

The IEC standard [268] does not specify which interval durations devices should support (i.e. possible values for T). Usually an integrating-averaging SLM lets the user choose from a range of durations, also called measuring times, e.g. from 1 second to 24 hours. Depending on the context or purpose, different durations may be preferred or mandated by various regulations. For example, assessments of exposure to noise in the workplace are generally conducted over 8 hour intervals, corresponding to a typical working day [96]. In that case the measured value indicates the constant SPL which over 8 hours would generate the same amount of sound energy as the SLM (and possibly the person who carried it) was actually exposed to due to sounds with varying SPL, generated by various sources (e.g. machines, vehicles, voices, etc.) throughout the day; see also section A.5.2.

To give real-time feedback while the interval is still running, the meter usually displays intermediate L_{eq} values, calculated over increasing durations. For example, when the meter is configured to measure L_{Aeq1h} and its numerical display has an update frequency of 1 Hz, then the user will first see a value that was computed over the first second, then over the first two seconds, etc., until the hour is over or the user aborts the session.

Integrating-averaging SLMs with logging support often have a "short L_{eq}" (or similarly named) mode, in which series of L_{eq} values are taken in succession over contiguous intervals of equal, short duration (down to $1/_8$ s) and stored in a digital memory. We hinted upon such a repeated L_{eq} measuring over equal intervals at the bottom of chart A.6. Short L_{eq} time histories have become the preferred method of storing or transmitting sound level data. This is mainly because after the measurements have been made, they can be used to calculate "overall L_{eq}" values over any sub-period of the series. Concretely, based on a sequence of n equivalent continuous sound level values ($L_{X\mathrm{eq},i}$) taken over contiguous intervals with possibly varying durations (t_i), the equivalent continuous sound level over a combined interval with duration $T = \sum_{i=1}^{n} t_i$ can be calculated as follows:

$$L_{X\mathrm{eq}T} = 10 \cdot \log_{10}\left(\frac{1}{T}\sum_{i=1}^{n} t_i \cdot 10^{\frac{L_{X\mathrm{eq},i}}{10}}\right) \quad [\mathrm{dB}(X)] \tag{A.26}$$

When all composite interval durations are equal this becomes:

$$= 10 \cdot \log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} 10^{\frac{L_{X\mathrm{eq},i}}{10}}\right) \quad [\mathrm{dB}(X)] \tag{A.27}$$

Which is analogous to the way we averaged SPL values with equation A.16 on page 274. For the computed overall L_{eq} value to conform to the IEC standard [268] each short L_{eq} value should too. In earlier times, overall L_{eq} values were computed by averaging of histories of time-weighted sound level values[55]. Because this meant results depended on the time-weighting used, short L_{eq} time histories are preferable.

[55] Basically this means using $L_{X\tau}(t)$ values as the exponent in equation A.26 or A.27.

A.5.1.4 Derived sound level descriptors

SLMs can usually indicate the *maximum* and sometimes also the *minimum* and even various *percentile sound levels*, derived from the sound level descriptors introduced above.

A.5.1.4.1 Maximum & minimum sound level

The *maximum* and *minimum (time-weighted) sound level* are respectively the highest and lowest value measured by an SLM over a period of time [268, 564]. They are based on the time-weighted sound level and are respectively denoted $L_{X\tau\mathrm{max}}$ and $L_{X\tau\mathrm{min}}$, where X is the frequency weighting and τ the time weighting. SLMs compute these values in real-time and usually they only apply to the running measuring session[56]. As an example, chart A.6 on page 302, shows the maximum and minimum of a Z-frequency weighted, F-time weighted sound level curve.

A.5.1.4.2 Percentile sound level

The *percentile sound level*, also known as *percentile-exceeded sound level* or *statistical sound level*, is denoted as L_n, where n is a number $\in [0, 100]$. Such descriptors indicate the sound level that is exceeded during n% of a given period of time [90, 564]. For example, L_{10} represents the sound level exceeded 10% of the time. There is no formal standard which defines percentile sound levels and therefore SLM manufacturers can choose to base it either on time-weighted or on time-average (typically short L_{eq}) sound level measurements. Some devices leave the choice to the user. Either way, to be able to compute percentile sound levels during or after a measuring session, the meter needs to keep a time history of the running session. Statistically speaking, an L_n value is the $(100-n)$th percentile of the data series. When calculated from time-weighted sound level data, the percentile sound level may be denoted as $L_{X\tau n}$ or L_{Xn}, where X is the frequency weighting and τ the time weighting. When based on time-average sound level values it is sometimes denoted as $L_{X\mathrm{eq}n}$. As an example chart A.6 shows three percentile sound levels of a Z-frequency weighted, F-time weighted sound level curve.

Percentile sound levels are often used to characterise the typical sound level, in a particular place and during a particular period of time, in a more detailed manner than what is possible with an overall L_{eq} value alone. For example, when assessing the sound level generated by traffic on a relatively busy road, the L_{90} value gives an idea of the *background noise*, i.e. the sound level that is still heard when there is a pause in the flow of traffic. The L_{10} value on the other hand, indicates the level that is surpassed only occasionally, e.g. when an ambulance or multiple heavy trucks are passing by.

[56] While less common, integrating-averaging SLMs may also indicate minimum and maximum $L_{X\mathrm{eq}T}$ values, as NoiseTube Mobile does for the $L_{\mathrm{Aeq,1s}}$ values measured during a track (see section 6.5.4.1).

A.5.1.5 Peak sound level

Apart from sound (pressure) level descriptors, some SLM devices can also measure the *peak sound level* (PSL), which is an expression in decibel of the *peak sound pressure*.

Peak sound pressure, denoted as p_{peak} and expressed in pascal, is defined as the greatest absolute instantaneous sound pressure during a stated time interval [268], as formalised by equation A.28:

$$\forall t \in [0, T] : p_{\text{peak}} \geq |p(t)| \tag{A.28}$$

The more practical peak sound level, denoted by $L_{X\text{peak}}$ (or $L_{X\text{pk}}$), is then defined as twenty times the logarithm to the base ten of the ratio of a peak sound pressure to the reference sound pressure ($p_0 = 20\,\mu\text{Pa}$), peak sound pressure being obtained with a standard frequency weighting [268]; as formalised by equation A.29:

$$L_{X\text{peak}} = L_{X\text{pk}} = 20 \cdot \log_{10}\left(\frac{p_{X\text{peak}}}{p_0}\right) \quad [\text{dB}(X)] \tag{A.29}$$

The de facto frequency weighting for PSL measurements is C-weighting, in which case the descriptor is denoted by L_{Cpeak} or L_{Cpk} and the unit is dB(C). No time-weighting is applied for peak measurements. Meters which support this feature indicate the PSL which occurred since the beginning of the measuring session. Whenever the value is surpassed later in the session, the displayed value is updated.

The difference between PSL (see equation A.29) and SPL (see equations A.5, A.23 and A.24) is that the former is based on the sound wave's absolute peak amplitude (i.e. the peak sound pressure; equation A.28), while the latter is based on its root-mean-square amplitude (i.e. the *effective sound pressure*; equation A.4). Thus, as it is based solely on a single amplitude spike, a PSL value is not representative for the sound level at a certain place or period of time. Still, PSL measurements are useful in specific cases, for instance to assess short, loud, banging sounds occurring in industrial contexts.

We should note that while they are easily (and often) mixed up, peak sound level and maximum sound level (see above) are very different measures. The latter is not an expression in decibels of the greatest absolute instantaneous sound pressure (i.e. p_{peak}), but rather the highest SPL that was measured, over a period of time. When both are simultaneously measured, the peak sound level will never be lower than the maximum sound level over the same period.

A.5.2 Dosemeters

A *dosemeter*, *dosimeter* (in American English), or *personal sound exposure meter* (PSEM) is a variant of the sound level meter which is worn on the human body. Such meters are intended to measure an individual's exposure to sound (or rather noise) over a period of time. They are mostly used in *occupational* settings as part of hearing damage prevention programs or regulations. Because the risk of hearing damage is related to both the level *and* the duration of exposure (see section A.3.2 and table A.1), PSEMs assess the cumulated effect of varying sound levels over a period of time, typically a working day or week. Usually the device is worn on the belt or carried in a pocket and has a wired microphone to be clipped on one's clothing, preferably close to one's ear, as shown in figure A.10a. Some recent types of PSEMs are small enough to be worn on the shoulder, eliminating the need for a possibly hindering wired microphone [374]; see figure A.10b.

Like SLMs, PSEMs have a long development history [468] and are the subject of various standards and regulations. The technical aspects of the devices are governed by the IEC 61252:2002 [267] and ANSI S1.25-1991 [16] standards. Other norms determine which sound level descriptors must be measured (i.e. which frequency-weightings, time-weightings and/or time intervals), how measurements are to be accumulated into an exposure quantity for a working day(/week), and under which limit that exposure should stay in order to be safe. In the EU ISO standard 1999:1990 [270] is followed, as stipulated by directive 2003/10/EC of the European Commission [174], while the U.S. follows its own rules [365, 544]. Because European and American regulations are not compatible many PSEMs have multiple operating modes [374].

(a) Dosemeter with wired microphone

(b) Shoulder-mounted dosemeter

Figure A.10: Personal sound exposure meters

While we will not go into the details of the exposure metrics stipulated by these regulations, it is interesting to note that the exposure sustained during a day, or up until a certain moment, can be expressed as a percentage of the maximum safe amount. This is called the *noise dose* (denoted by D) and it is the origin of the term dosemeter/dosimeter. Because the meter tracks the dose throughout the day, as shown on the display of the device in figure A.10a, a single glance at the device gives the wearer a clear idea of his or her exposure and the associated risk – i.e. if you reach 100% before the day is over, you know your hearing is at risk. Despite these simple semantics, noise dose is no longer used as the primary metric on recent PSEMs. The reason is that if exposure is only expressed as a percentage, instead of a physical quantity, devices would become obsolete whenever regulations change.

A.6 Conclusion

This appendix was devoted to the physical phenomenon of sound. First we explained what sound really is, then we focused on how humans perceive it, how it can be represented by audio signals and how sound, or rather sound (pressure) level, can be measured.

The most important things to remember are:

- sound travels through a medium as a wave of local/temporal *pressure deviations*;
- *sound waves* are described by their *amplitude* and *wavelength* or *frequency*;
- *sound pressure* is a measure of amplitude, it represents the pressure deviation from the ambient (air) pressure caused by a sound wave, it is expressed in pascal;
- *sound (pressure) level* is the ratio of the *effective sound pressure* (RMS) to the *reference sound pressure*, expressed in decibels;
- *human hearing* has a *frequency range* of $\pm 20\,\text{kHz}$ and a *dynamic range* of $\pm 130\,\text{dB}$;
- humans are not equally sensitive to all frequencies: the *loudness* of sounds depends on the sound pressure and the frequencies present in the sound (and other factors);
- sound waves can be converted into *analogue audio signals* by means of a microphone and a loudspeaker converts such a signal back into audible sound;
- the *frequency response* of microphones (and loudspeakers) is not necessarily flat;
- analogue audio signals can be converted into *digital audio signals* with an *ADC*;
- the main properties of a digital audio signal are its *sampling rate* and its *bit depth*;
- digital audio can be stored and transmitted in various *formats*, which are categorised as *encodings* and *containers*;
- sound (pressure) level can be measured with a *sound level meter* (SLM);
- SLMs compensate (albeit imperfectly) for the *frequency response* of human hearing by applying *frequency weighting*, the most commonly used way is *A-weighting*;
- SLMs are divided in two kinds, *conventional* ones and *integrating-averaging* ones, which differ in the way they average sound pressure over time. This means there are also two kinds of *sound level descriptors*: *time-weighted sound-level* and *time-average sound level* (also known as *equivalent continuous sound level* or "L_{eq}");
- from the point of view of non-professional users, standard-compliant SLMs, especially integrating-averaging ones, are considerably expensive;
- *dosemeters* or *personal sound exposure meters* (PSEMs) measure the exposure to sound of an individual person over a period of time, typically a working day.

While they are commonly interchanged and even confounded, the terms *sound* and *noise* represent clearly distinct concepts. *Sound* is a neutral term that applies to all audible vibrations (see section A.2). *Noise*, on the other hand, is generally a subjective term that humans use to label sounds as *unwanted*. Because the focus was on sound, we have tried to avoid using the word “noise” in this appendix[57]. In chapter 4, we turn our attention to the subject of noise and the problems it causes.

[57] For instance, we have refrained from using terms such as *noise (level) meter*, which is popular but essentially incorrect because devices such as SLMs cannot make the “human” distinction between wanted and unwanted sound and consequently measure the level of *all* sounds.

Appendix B

Precision, accuracy & calibration

In this dissertation we frequently discuss the topic of data quality and more specifically the precision and accuracy of (sound level) measurements. This appendix provides a short recap of high school physics on the distinction between precision and accuracy, different types of measurement errors and how they can be remedied by averaging or calibration.

B.1 Precision vs. accuracy

While commonly interchanged in everyday speech the terms *accuracy* and *precision* have distinct meanings. To understand the difference we need to take a closer look at measurement errors. Regardless of what[1] is being measured or the device being used, measurement errors can be split in a *random* and a *systematic* component.

Random errors are inherently unpredictable differences between repeated measurements of a constant attribute or quantity. Affected measurements are scattered about the true value and the errors tend to have an arithmetic mean of zero. Causes of random errors include fluctuations in the measurement apparatus itself or in the user's interpretation of the readings. Usually random errors are remedied by simply averaging a bunch of readings to produce one value (i.e. the variations, or "noise", is "averaged out").

Systematic errors – also called *biased errors* or just *biases* – cause predicable (as in consistent) deviations from the true value of the measured attribute or quantity. As opposed to random errors, systematic errors cannot be averaged out (i.e. they remain present in the mean) and are typically remedied by means of *calibration* instead (see below).

[1] For instance distance, weight, sound pressure level, etc.

The smaller variability (standard deviation) of repeated measurements (i.e. the lower the random error), the higher the *precision* of measurement. The smaller the difference between the arithmetic mean[2] of repeated measurements and the true value (i.e. the lower the systematic error), the higher the *accuracy* of measurement.

In other words, precision refers to the "exactness of measurement" whereas accuracy refers to its "correctness". More formally, precision is a measure of *spread* and accuracy is a measure of *bias*. Figure B.1 illustrates the difference by means of the analogy of arrows being shot at a target. As shown it is possible for data to be inaccurate but precise, or accurate but imprecise.

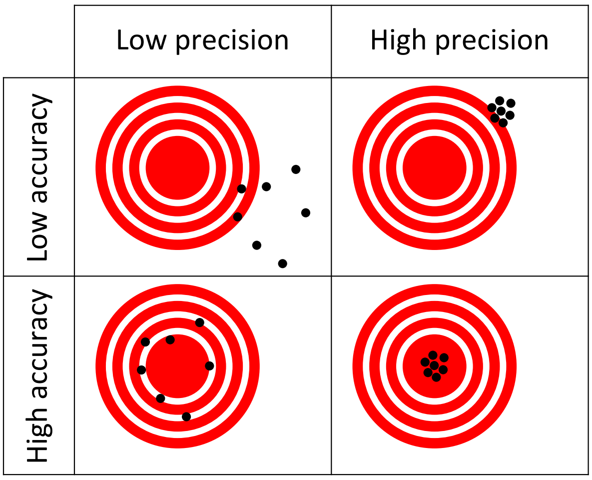


Figure B.1: Precision versus Accuracy

As a concrete example, consider an object with known weight that is repeatedly weighted using the same scale device. Suppose we find that the differences between the measurements are within 0.25 kg, but all measurements consistently overestimate the object's true weight by 2 kg. Then the 0.25 kg margin of random errors tells us something about (*im*)*precision*, and the systematic error of 2 kg w.r.t. the true weight is an indication of (*in*)*accuracy*.

B.2 Calibration

Calibration is a comparative procedure in which the accuracy of a measuring device is evaluated and improved – i.e. detecting and correcting for systematic errors – by comparing measured values with expected ones. In the example given above the scale can be calibrated by weighting objects with known weights – or alternatively, weighting objects with unknown weight on both the scale being calibrated and a second, trusted "reference scale" – in order to calculate a calibration offset, in this case: subtract 2 kg. This offset can than be applied to all subsequent measurements to correct for the systematic error.

In the case of our example the systematic error is constant (independent of the weight the scale always overestimates by 2 kg). However systematic errors can also be proportional to the true value and/or change in response to external conditions (e.g. temperature). Then one must devise a more complex *calibration function* which computes adjustments that depend on the measured value (e.g. involving a scale factor) and/or external conditions.

[2] I.e. after correction for random errors.

Appendix C

The NoiseTube Prototype

Between June and September 2008 a first prototype for the NoiseTube system was developed at Sony CSL Paris. This prototype was not functionally complete and was only tested internally at. Almost no code was carried over to the current NoiseTube system. Nevertheless there are a few noteworthy aspects, especially regarding the functionality for the tagging functionality.

C.1 Architecture

The architecture of the NoiseTube prototype consisted of 3 software components: a portable digital audio recorder, a smartphone application called *Mobile Noise Tagger* (MNT), a desktop application called *NoiseTube Uploader*, and a web application running on a central server. Figure x shows an overview of the architecture. The diagram in figure C.1 shows how these components interacted.

As opposed to the current system (see section 5.4) the NoiseTube prototype did not employ phones as actual noise sensors. Instead, we users had to carry a portable digital audio recorder to continuously record audio to file. This meant that the measuring of the sound pressure level was effectively delayed to a later stage.

From the onset of the NoiseTube project the goal was always to enable the measuring of sound level on the mobile phone itself. The reasoning behind this different architecture was twofold. On the one hand we were unsure whether the smartphones of back then would be capable of running a real-time sound level measuring algorithm, hence we want to prototype that on a desktop instead. On the other, this architecture gave us the

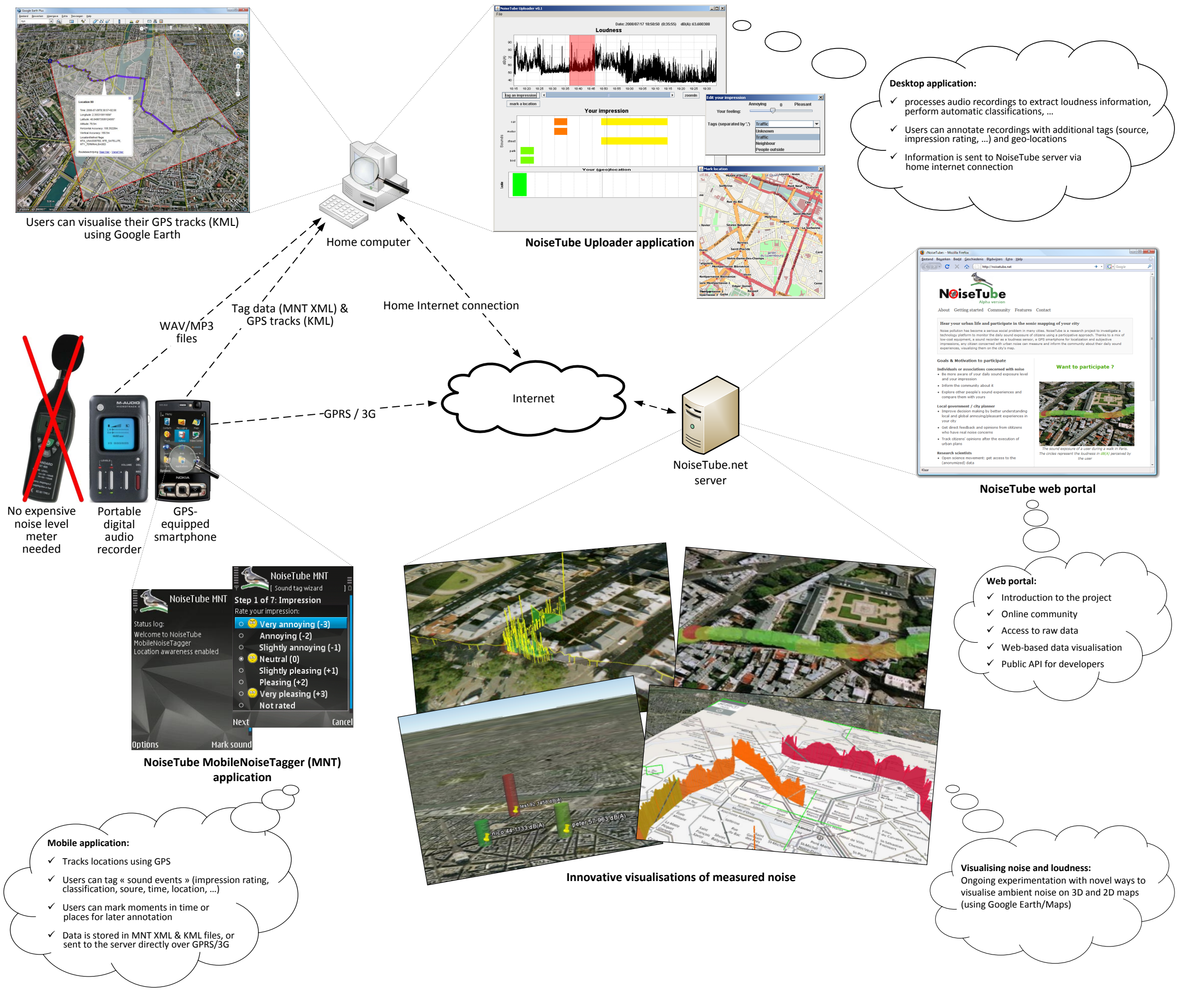


Figure C.1: Architecture of the NoiseTube Prototype (September 2008)

freedom to prototype different aspects and components of the NoiseTube system in relative isolation, while gaining experience with the different technologies.

C.2 NoiseTube Mobile Noise Tagger

The NoiseTube Mobile Noise Tagger (MNT) served two principal purposes: tracing the user's location by means of GPS and allowing him/her to tag sounds.

In fact the 7-step "sound tag wizard" was much more than a social tagging system. However, it was way too time consuming to be practical while on the go.

The 9 sound source categories were taken from [230]. We presented them to the user in a randomised order.

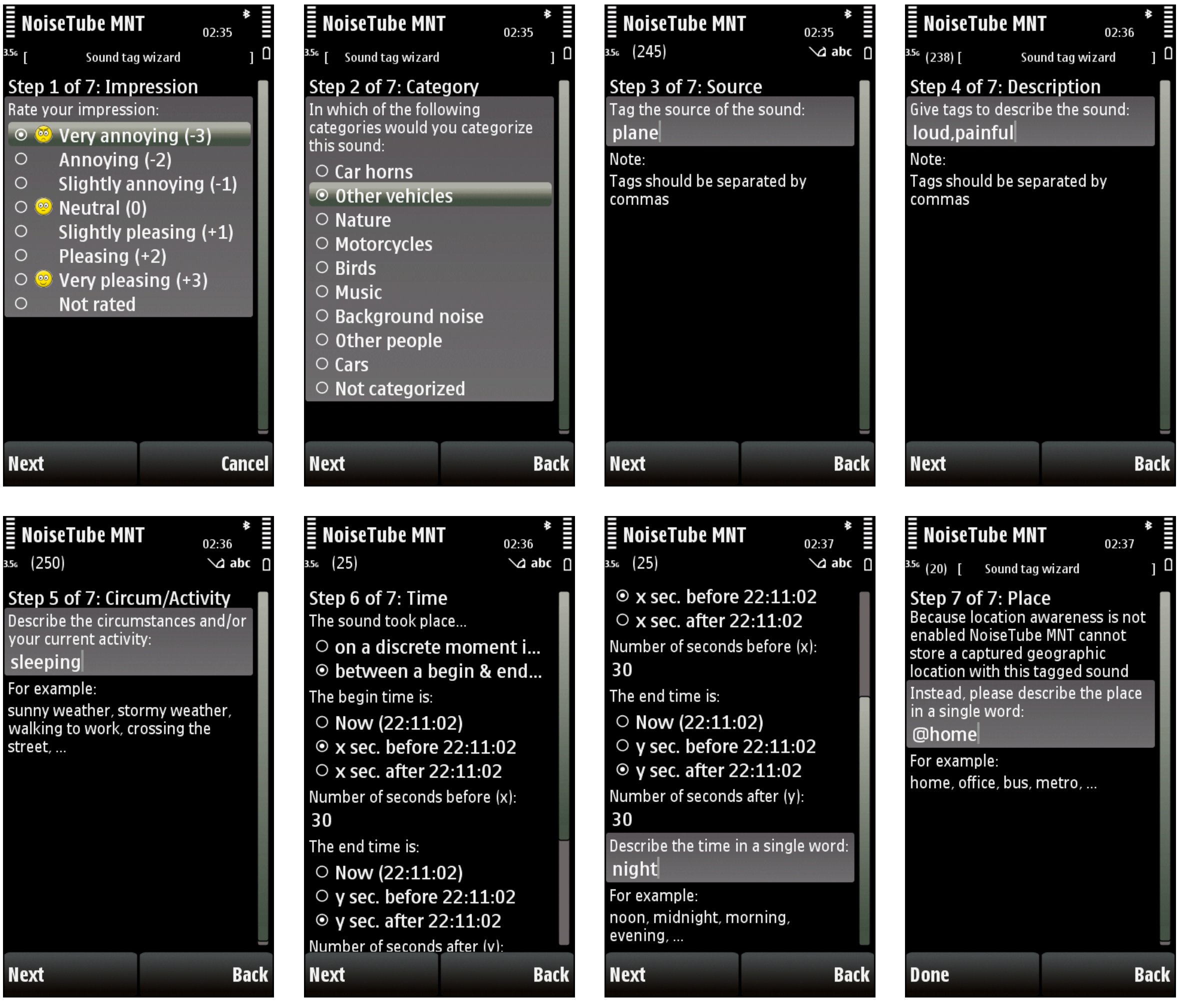


Figure C.2: The 7-step *Sound tag wizard* of Mobile Noise Tagger

C.3 NoiseTube Uploader

The NoiseTube Uploader application allowed users to combine the data collected on their phone (consisting of a series of time stamped geographical coordinates and "sound event" descriptions) with the actual sound recorded using the digital audio recorder.

The desktop application would take the audio recording (stored as a WAVE file) and the data file (stored as an XML file) and present the user with a visualisation of both. The audio file was analysed to extract a series of $L_{\mathrm{Aeq,1s}}$ measurements, which were then combined with (geo)tags by matching of timestamps.

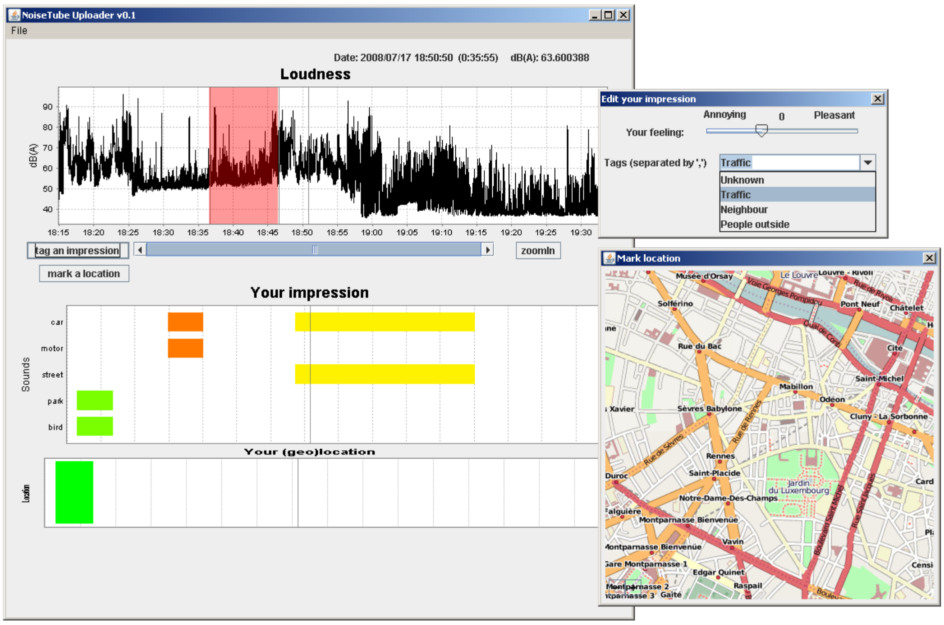


Figure C.3: NoiseTube Uploader desktop application

Appendix D

NoiseTube Data Interchange Specifications

Here we document the data interchange APIs and file formats of the NoiseTube system.

D.1 Web API

All direct communication between the NoiseTube Mobile app and the NoiseTube Community Memory (CM) happens through the HTTP protocol. To this end, the Community Memory exposes a simple API which is documented in table D.1:

Command [HTTP request type]	URL parameters		Request body	Response body [HTTP status code]	
	Required	Optional		Success	Failure
`ping` [GET]	-	-	-	"ok" [200]	-
`mobilecalibrations` [GET]	-	-	-	calibrations list in XML format (see below) [200]	-
`authenticate` [GET]	`login` (username), `password`	-	-	API key of the user [200]	"error"
`newsession` [GET]	`key`	`client`, `clientversion`, `devicebrand`, `devicemodel`	-	"ok <TrackID>" [200]	error message
`resumesession` [GET]	`key`	`track` (track ID)	-	"ok <TrackID>" [200]	[404]
`update` [GET]	`key`, `time`, `db` (sound pressure level), `l` (location: coordinates/tag)	`track` (track ID), `tag` (comma-separated user tags), `autotag` (comma-separated automatic tags)	-	"ok" [200]	error message
`upload` [GET]	`key`	`track` (track ID)	Batch of measurement and tagged intervals encoded as JSON (see below)	"ok" [200]	error message
`taginterval` [GET]	`key`, `beginIndex` (index of first measurement), `endIndex` (index of last measurement)	`track` (track ID), `tag` (comma-separated user tags), `autotag` (comma-separated automatic tags)	-	"ok" [200]	error message
`endsession` [GET]	`key`	`track` (track ID)	-	"ok" [200]	[404]
`search` [GET]	`key`	`user` (ID or username), `tag`, `city` (ID), `geo` (geographical bounding box), `dbmax`, `dbmin`, ...	-	Measurements as JSON [200]	error message

Table D.1: Web API exposed by NoiseTube Community Memory

All commands (listed in the first column) are passed in the URL of an HTTP GET or POST request. The base URL for all commands is http://noisetube.net/api/. For instance, to test if the Community Memory can be reached and functions correctly NoiseTube Mobile will issue a GET request with as URL: http://noisetube.net/api/ping.

Most other commands are only accessible to registered users. To control access these commands require the key parameter to be passed. The API key is a 160-bit number that uniquely identifies the user and is generated upon account creation. Users can find it on their profile page. However, they do not need it to use NoiseTube Mobile, since the app only asks for their username and password. To verify the combination the app uses the authenticate command. If correct the CM answers with the user's API key, which is then stored[1] by the app and passed as a parameter with subsequent commands. For instance, when a new track is started (and submission to the CM is activated) the app will issue a GET request with as URL: http://noisetube.net/api/newsession?key=<APIKey>. When measurements and tagged intervals are being sent in real-time (see sections 6.3.5 and 6.5.4.6), the app respectively uses the update and taginterval commands. When they are sent as a batch the upload command is used. This is the only command that uses a HTTP POST request, the body of which contains the batch of measurements and tagged intervals encoded as JSON [104], as shown by the example in box D.1:


```
{
    "measures": [
        ["2012-02-02T14:11:29+01:00",56,"geo:51.03412455,3.71387497","music,relaxing",""],
        ["2012-02-02T14:11:30+01:00",44,"geo:51.03412395,3.71387494","",""],
        ["2012-02-02T14:11:31+01:00",33,"geo:51.03409632,3.71394105","",""],
        ["2012-02-02T14:11:32+01:00",33,"geo:51.03408499,3.71398757","",""],
        ["2012-02-02T14:11:33+01:00",33,"geo:51.03205285,3.71037704","",""],
        ["2012-02-02T14:11:34+01:00",33,"geo:51.03406953,3.71408881","",""],
        ["2012-02-02T14:11:35+01:00",35,"geo:51.03399937,3.71435226","",""],
        ["2012-02-02T14:11:36+01:00",57,"geo:51.0339944,3.714418127","","variation:high"],
        ["2012-02-02T14:11:37+01:00",39,"geo:51.03397431,3.71449862","",""],
        ["2012-02-02T14:11:38+01:00",45,"geo:51.03394482,3.71457105","",""],
        ["2012-02-02T14:11:39+01:00",36,"","",""],
        ["2012-02-02T14:11:40+01:00",40,"","",""],
        ["2012-02-02T14:11:41+01:00",35,"","",""],
        ["2012-02-02T14:11:42+01:00",52,"","",""],
        ["2012-02-02T14:11:43+01:00",57,"","",""],
        ["2012-02-02T14:11:44+01:00",36,"","",""],
        ["2012-02-02T14:11:57+01:00",34,"","",""],
        ["2012-02-02T14:11:58+01:00",34,"","",""],
        ["2012-02-02T14:11:59+01:00",43,"","",""],
        ["2012-02-02T14:12:00+01:00",36,"","",""],
        ["2012-02-02T14:12:01+01:00",45,"","",""],
        ["2012-02-02T14:12:02+01:00",48,"","",""],
        ["2012-02-02T14:12:03+01:00",35,"","",""],
        ["2012-02-02T14:12:04+01:00",51,"","","variation:high"],
        ["2012-02-02T14:12:05+01:00",34,"","",""],
        ["2012-02-02T14:12:06+01:00",33,"","",""],
        ["2012-02-02T14:12:07+01:00",39,"","",""],
        ["2012-02-02T14:12:08+01:00",33,"","",""],
        ["2012-02-02T14:12:09+01:00",34,"","",""],
        ["2012-02-02T14:12:10+01:00",44,"","",""]
    ],
    "taggedIntervals": [
        [284,312,"quiet,@home",false]
    ]
}
```


Individually tagged measurement (by user)

GPS reception lost

Individually tagged measurements (by app)

Box D.1: Example of a batch of 30 measurements and 1 tagged interval in JSON format

[1] The API key is stored persistently such the user does not have to log-in again the next time.

The `search` command is not indented for use by NoiseTube Mobile (although it could be). Instead it provides interested parties with a simple interface through which they can manually or programmatically query the measurements in the Community Memory database (with a maximum of 500 measurements returned every time). For instance, to get measurements of 70 dB(A) or higher tagged with the word "loud" one would use the following query: http://noisetube.net/api/search?key=<APIKey>&dbmin=70&tag=loud.

More information about the Web API, including some additional examples, is given here: http://noisetube.net/api_overview.

D.2 Track file format

When data is being saved to locally (see sections 6.3.5 and 6.5.4.6) every track is stored in a separate XML file created on the phone's built-in memory or removable memory card. These track files can then later be manually uploaded to the Community Memory using any Web browser. The format of these files is documented by example in box D.2:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<NoiseTube-Mobile-Session startTime="2012-01-23T01:21:56+01:00"
  client="NoiseTubeMobileAndroid" clientVersion="v1.3.0" clientBuildDate="2012-01-23 00:55:44 CET"
  deviceBrand="HTC" deviceModel="Vision" devicePlatform="Android" devicePlatformVersion="10"
  calibration="Calibration for HTC HTC Hero (source: downloaded from NoiseTube.net)" credibility="G">
    <measurement timeStamp="2012-01-23T01:21:56+01:00" loudness="36"/>
    <measurement timeStamp="2012-01-23T01:21:57+01:00" loudness="50"/>
    <measurement timeStamp="2012-01-23T01:21:58+01:00" loudness="72" autotags="variation:high"/>
    <measurement timeStamp="2012-01-23T01:21:59+01:00" loudness="76"/>
    <measurement timeStamp="2012-01-23T01:22:00+01:00" loudness="83" tags="loud,plane,@home"/>
    <measurement timeStamp="2012-01-23T01:22:01+01:00" loudness="75"/>
    <measurement timeStamp="2012-01-23T01:22:02+01:00" loudness="63"/>
    <measurement timeStamp="2012-01-23T01:22:03+01:00" loudness="55"/>
    <measurement timeStamp="2012-01-23T01:22:04+01:00" loudness="60"/>
    <measurement timeStamp="2012-01-23T01:22:05+01:00" loudness="62"/>
    <measurement timeStamp="2012-01-23T01:22:06+01:00" loudness="60"/>
    <measurement timeStamp="2012-01-23T01:22:07+01:00" loudness="61" location="geo:51.03412395,3.71387494"/>
    <measurement timeStamp="2012-01-23T01:22:08+01:00" loudness="62" location="geo:51.03409632,3.71394105" tags="music,@home"/>
    <measurement timeStamp="2012-01-23T01:22:09+01:00" loudness="57" location="geo:51.0340849,3.71398757"/>
    <measurement timeStamp="2012-01-23T01:22:11+01:00" loudness="61" location="geo:51.03407449,3.71403766"/>
    <measurement timeStamp="2012-01-23T01:22:12+01:00" loudness="65" location="geo:51.03406953,3.71408881"/>
    <measurement timeStamp="2012-01-23T01:22:13+01:00" loudness="69" location="geo:51.03405031,3.71417284"/>
    <measurement timeStamp="2012-01-23T01:22:14+01:00" loudness="67" location="geo:51.03402698,3.71423406"/>
    <measurement timeStamp="2012-01-23T01:22:15+01:00" loudness="74" location="geo:51.03401106,3.71428318"/>
    <measurement timeStamp="2012-01-23T01:22:16+01:00" loudness="77" location="geo:51.03399937,3.71435226"/>
    <measurement timeStamp="2012-01-23T01:22:17+01:00" loudness="81" location="geo:51.0339944,3.714418127"/>
    <measurement timeStamp="2012-01-23T01:22:18+01:00" loudness="84" location="geo:51.03397431,3.71449862"/>
    <measurement timeStamp="2012-01-23T01:22:19+01:00" loudness="82" location="geo:51.03394482,3.71457105"/>
    <measurement timeStamp="2012-01-23T01:22:20+01:00" loudness="88" location="geo:51.03391905,3.71465777"/>
    <measurement timeStamp="2012-01-23T01:22:21+01:00" loudness="92" location="geo:51.03386798,3.71479803"/>
    <measurement timeStamp="2012-01-23T01:22:22+01:00" loudness="88" location="geo:51.03386750,3.71479899"/>
    <measurement timeStamp="2012-01-23T01:22:23+01:00" loudness="85" location="geo:51.03385382,3.71486124"/>
    <measurement timeStamp="2012-01-23T01:22:24+01:00" loudness="80" location="geo:51.03384791,3.71495656"/>
    <measurement timeStamp="2012-01-23T01:22:25+01:00" loudness="78" location="geo:51.03384284,3.71500853"/>
    <measurement timeStamp="2012-01-23T01:22:26+01:00" loudness="74" location="geo:51.03384264,3.71506913"/>
    <measurement timeStamp="2012-01-23T01:22:27+01:00" loudness="68" location="geo:51.03383282,3.71519023"/>
    <measurement timeStamp="2012-01-23T01:22:28+01:00" loudness="64" location="geo:51.03383274,3.71519091"/>
    <measurement timeStamp="2012-01-23T01:22:29+01:00" loudness="61" location="geo:51.0338282,3.71524196"/>
    <measurement timeStamp="2012-01-23T01:22:30+01:00" loudness="64" location="geo:51.03383919,3.71530209"/>
    <measurement timeStamp="2012-01-23T01:22:31+01:00" loudness="56" location="geo:51.0338481,3.71536325"/>
    <measurement timeStamp="2012-01-23T01:22:32+01:00" loudness="58" location="geo:51.03385966,3.71542322"/>
    <taggedInterval beginIndex="19" endIndex="31" tags="loud,cars,trucks"/>
    <measurement timeStamp="2012-01-23T01:22:33+01:00" loudness="56" location="geo:51.03387576,3.71549197"/>
    <measurement timeStamp="2012-01-23T01:22:34+01:00" loudness="60" location="geo:51.03388411,3.7155487"/>
    <!-- ... -->
</NoiseTube-Mobile-Session>
```

NoiseTube Mobile app info.
Device hard/software info.
Calibration info.
Individually tagged measurement (by app)
Individually tagged measurements (by user)
1st GPS fix
taggedInterval applies to

Box D.2: Example of a track XML file

We intentionally kept the format as simple as possible such that users can make sense of the data when they want to analyse it "manually", and to allow even novice programmers to write a parser in order to use the track files for whatever purpose. For the same reason we chose not to base the track file format on an existing standard[2] which would likely cause more overhead and could reduce readability for end-users.

D.3 Calibrations file format

As discussed in section 6.5.4.2 we have introduced a simple, human-readable and -editable file format for interchanging and distribution of calibration settings for NoiseTube Mobile. This XML-based format, along with the "credibility" values, is documented in box D.3:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<calibrations lastChanged="2011-11-13">
      <!-- credibilityIndex values:
              - B: internal (=SonyCSL/VUB) professional in ideal conditions (verified)
              - C: internal professional (verified)
              - D: internal professional (unverified)
              - E: external professional (unverified)
              - F: end user (unverified)
            Not to be used here, but used internally in NoiseTube Mobile apps:
              - A: same as B but specific to a single physical instance of a device model
              - G: brand match, model mismatch
              - H: brand and model mismatch (default used)
      -->
      <!-- Nokia -->
      <calibration deviceBrandID="2" deviceBrand="Nokia" deviceModel="5230" credibilityIndex="B" overallDefault="true" brandDefault="true">
           <creator>BrusSense-VUB</creator>
           <comment>
                 Values obtained by 10th degree polynomial regression of the calibration points of 11 separately
                 calibrated Nokia 5230 devices. Calibration done by Ellie D'Hondt in summer-autumn 2010 in an
                 anechoic chamber at the VUB. Regression done by Matthias Stevens in November 2011.
           </comment>
           <correction input="24.186232000222276" output="30.00000119605167"/>
           <correction input="25.275949001342457" output="35.000001181492784"/>
           <correction input="26.68680900279276"  output="40.000001006570955"/>
           <correction input="28.784411004949003" output="45.00000087417067"/>
           <correction input="33.424043004659126" output="50.00000027302083"/>
           <correction input="42.25612598236026"  output="54.99999997813444"/>
           <correction input="49.31082896454885"  output="59.99999987210208"/>
           <correction input="58.9337909402532"   output="64.9999998414196"/>
           <correction input="66.33700592156188"  output="69.9999995701437"/>
           <correction input="71.53530390843744"  output="75.0000002167144"/>
           <correction input="76.44825889603342"  output="80.00000046478817"/>
           <correction input="80.82507288498303"  output="85.0000005311158"/>
           <correction input="83.92979787714435"  output="90.00000064296182"/>
           <correction input="86.23962087131261"  output="95.00000048807124"/>
           <correction input="88.26074686620976"  output="99.99999888468301"/>
           <correction input="90.6186978602565"   output="104.90000009140931"/>
     </calibration>
     <!-- ... -->
</calibrations>
```

Box D.3: Fragment of the NoiseTube Mobile calibration settings file

The "master" `calibration.xml` file can be downloaded from the NoiseTube website at the following address: http://noisetube.net/calibrations.xml.

It can also be requested using the `mobilecalibrations` command of the Web API, which is how NoiseTube Mobile downloads it.

[2] Such as EEML [244] or KML [298], both of which are XML-based.

Appendix E

All about platforms

This appendix provides background information on the hardware (i.e. smartphones) and software (e.g. Java ME and Android) platforms we have targeted with NoiseTube Mobile. Apart from describing technical aspects, we also spend attention on the evolutions in the market for smartphones and the software platforms that run on them. Moreover, based on our experiences we also present generalised guidelines regarding cross-platform application development for Java ME and Android.

E.1 Smartphones & co

In this dissertation, we wrote a lot of exciting things about smartphones, but what exactly are these? And what about other phones?

Put simply, smartphones can be defined as high-end mobile phones with advanced computing abilities. Yet it is difficult to support that definition with a list of specific hard- and software technologies or abilities that a device is required to have to deserve the smartphone label. There are two reasons for this.

First, due to constant technological innovation and market effects, such a list would be ever changing. As new features are introduced in the latest, most expensive smartphones, older features trickle down to cheaper models, including devices that are not regarded as smartphones, although they would have been just a couple of years before.

Second, what constitutes a smartphone has arguably as much to do with the needs and usages the devices have come to fulfil in people's lives – often by replacing other devices in the process – than it has with purely technical aspects. The first smartphones were born by integrating the functionality of the archetypical, late-1990s mobile phone[1] with that of the *personal digital assistant* (PDA), a type of device that has all but disappeared now. Since then, smartphones have, to varying degrees, replaced other devices such as portable media players, digital photo and video cameras and satellite navigation units. Every time, this has been a process of both integration and improvement: the functionality of an existing device was adopted but also augmented, usually by leveraging the Internet to provide personalised, social or location-based services and content. The next device that smartphones, along with tablet computers[2], are starting to replace (to some extent) is the desktop or laptop PC [144].

A current list of hardware specifications for high-end smartphones would include: a high-resolution touch screen, one or two high-resolution cameras, one or two microphones, a fast CPU (often dual- or even quad-core), about 1 GB of RAM, at least 16 GB of storage memory, a GPS receiver, several additional sensors, high-speed data access via Wi-Fi and mobile broadband (3G/4G), and local connectivity features such as Bluetooth and NFC. On the software side, smartphones are defined by the capability to run a wide variety of "apps", usually distributed through an "app store" service. Typically smartphones have an OS with multitasking support, allowing to run multiple apps simultaneously.

In recent years, smartphones have come to make up a significant and growing share of the mobile phones in use worldwide[3]. This is because they have become both more desirable and more affordable. Still, prices of the most advanced models have hardly dropped, even to the contrary[4]. However, the overall range of (what are called) smartphones has diversified and broadened towards lower price points, often helped by carrier subsidies[5]. Moreover, because features originally associated with expensive smartphones (e.g. touch-screens) later appear in cheaper devices – whether they are called smartphones or not – it could be argued that at smartphones are becoming cheaper "at feature-parity".

Cheaper mobile phones that do not meet contemporary smartphone expectations are typically grouped in two categories: *feature phones* and *"dumb" phones*. Feature phones include many smartphone functionalities but are less capable in terms of hardware and software, although they generally do support user-installable apps. In many ways they resemble the smartphones of a couple of years before. But the difference is often vague:

[1] Which at that time had just three principal functionalities (i.e. making phone calls, sending/receiving text messages and storing the phone numbers of contacts) and only a limited set of additional features (e.g. a clock, an alarm, a calculator and a couple of simple games such as Nokia's infamous *Snake*).

[2] Which, despite having a bigger form factor and often lacking phone functionality, are very similar to smartphones in terms of hard- and software.

[3] From January 2008 up to and including September 2011 an estimated 932 million smartphones have been sold, accounting for about 18 % of all mobile phone sales [200–204].

[4] Which may help to explain why some people have adopted these gadgets as kind of a status symbol.

[5] *Carrier subsidies* are arrangements in which network operators (also called *carriers*) offer devices at a discount, or even for free, to subscribers that sign up for a binding (multi-year) plan. This is common in most countries but not in Belgium due to legal restrictions that were only fairly recently lifted [120].

what one vendor calls a feature phone another may well call an "entry-level" smartphone. Dumb phones, on the other hand, are still cheaper and often physically compact mobile phones, that besides archetypical features may also include things like a low-resolution camera and ditto colour screen, Bluetooth and simple media player functionality.

Besides their lower price feature phones and dumb phones have the edge on smartphones in another aspect as well: autonomy. So far, innovations in rechargeable battery technology have been unable to offset the energy requirements of smartphones' "advanced computing abilities" and the intensive usage thereof. Dumb phones can often be used for up to a week, i.e. to regularly make phone calls and send or receive text messages. For smartphones, which are used for those as well as many other things, it is not uncommon to run out of juice after a single day[6].

It is important to note that, as new features are introduced at the high-end of the market, many functions once associated with expensive smartphones trickle down to cheaper ones, feature phones and even "dumb" phones (also see below).

E.2 The market for smartphones & their platforms

The smartphone market – as well as the very definition of what constitutes a smartphone (see above) – probably changes quicker than any other consumer electronics segment. In the last decade no single vendor has been able to remain unchallenged for long. In parallel with the rapidly shifting market shares of the device vendors there is fierce competition among the software platforms which run on the devices. Most manufacturers have switched platforms several times and many are hedging their bets by putting out multiple products powered by different platforms. Apart from the hardware manufacturers several large software companies (e.g. Google, Microsoft and Oracle) have meddled in this struggle in pursuit of a piece of the lucrative market of mobile apps and services.

Below we take a look at the situation in the market at in mid-2008 and how it has evolved since then. We take the perspective of mobile application developers. Therefore our main focus is on the software platforms, rather than on brands and vendors.

E.2.1 The situation in mid-2008

In mid-2008, when the NoiseTube project started, the global and European smartphone markets were dominated by Nokia. To illustrate the then balance of power, let's take

[6] Especially models with a large screen, which is typically the most power-hungry hardware component.

a look at the market shares of Nokia and its main competitors during Q2 of 2008, as reported [76] by market research firm Canalys[7]. In that quarter 45.5 % of worldwide smartphone sales concerned Nokia devices and the company produced 71.2 % of the units sold in EMEA. Nokia's main rival at the time was Research in Motion (RIM), the outfit behind the BlackBerry brand, which held 16.7 % of the global market and 7.2 % of the EMEA market. A year had passed since Apple unveiled its iconic iPhone to worldwide media attention, but at a modest 2.1 % share of global sales the company's attack on the smartphone market was still in a premature stage, although that was about to change.

Almost all Nokia smartphones run on Symbian OS [519, 592], an operating system originally developed by Symbian Ltd[8]. Besides Nokia, many other manufacturers – most notably Samsung, Motorola and Sony Ericsson – have licensed Symbian OS to use it in some of their product lines [579]. Still according to Canalys [76], an estimated 58.2 % of smartphones sold in the world during Q2 2008 ran Symbian OS.

We should note that there are visual as well as technical differences between Symbian OS devices of different vendors. Because the OS itself lacks a user interface and applications (e.g. for basic telephony features) an additional software layer needs to be put on top of it. The most common example is the Nokia-developed S60 user interface and application platform [591], which has shipped on virtually all Nokia's Symbian OS devices and on some from other vendors (with a changed visual appearance).

At the time there were two main approaches to the development of applications for Symbian OS: one could write a *native* application in Symbian C++[9] or one could program a so-called *MIDlet* in Java. MIDlets are applications for the Java ME CLDC/MIDP middleware platform (see section E.3). Native applications are called that way because they run directly on the operating system and CPU, while MIDlets – like other Java programs – are interpreted by a virtual machine which is itself a native application. Because they run "closer to the metal" native applications have the advantage that are generally faster than MIDlets and can interact with more device-specific features.

Programming in Symbian C++ has a steep learning curve and deploying native applications can be cumbersome due to incompatibilities between different Symbian OS versions and the different user interface layers (and versions thereof) vendors ship on top of it[10]. Developing MIDlets is generally less complicated and there are fewer compatibility or portability issues due to the fact they are executed by a virtual machine. Moreover,

[7] We should note that Canalys bases these percentages on estimated sales figures.

[8] Symbian Ltd. was established in 1998 by Nokia, Ericsson, Psion, Matsushita (the company behind the Panasonic brand) and Motorola. Over the years ownership changed frequently as Symbian OS licensees bought or sold stakes in the firm. Nokia gradually extended its stake and became the sole owner when it acquired all remaining shares in December 2008 [593].

[9] A non-standard variant of C++.

[10] Native Symbian applications often need to be recompiled or even adapted in order to work on different devices from multiple or even the same manufacturer.

MIDlets can run on other mobile operating systems as well. For example, they are supported on RIM's BlackBerries and on many non-Symbian OS devices which were sold at the time by companies like Sony Ericsson, Samsung and Motorola[11]. Consequently the Java ME CLDC/MIDP platform had (and has) an even bigger market share than Symbian OS itself and it was the most popular mobile application platform at the time.

E.2.2 Evolution up to now

In the years since the start of the NoiseTube project, the smartphone market has evolved at a spectacular pace. The biggest impacts were caused by the emergence of Apple's iPhone and Google's Android platform, which thoroughly shook up the balance of power, especially in the high-end and mid-level range of the market.

With its large touchscreen and simplicity of use the iPhone [29] was a generation ahead of the other smartphones on the market in 2007-'08. Thanks to that, its pretty design and the sort of well-orchestrated media hype Apple is known for, it became very popular despite its high price. Initially global sales grew slowly due to limited availability outside the US[12], but once that was sorted the iPhone took the high-end range of the market by storm. In 2008, Apple's share of the market for US$ 300+ phones was at 25 %, by 2010 it had jumped to 61 %[13]. The iPhone also introduced the general public to mobile applications, now often simply called *apps*. The success of the iTunes App Store [31, 32], Apple's distribution and sales channel for iOS[14] apps, almost made some commentators forget that user-installable mobile applications had been around for many years.

In late 2008, a partnership headed by Google took up the challenge and introduced Android [226] (see section E.4), which was the first, and possibly still the only, mobile device platform to match iOS in terms of features and ease of use. While being the driving force behind the platform, Google itself does not manufacture Android devices[15]. Instead, the software is shipped on products manufactured by a growing group of companies, many of which previously licensed Symbian OS and/or used proprietary platforms. The choice of Android devices on the market is big and spans a wide price range.

[11] Nokia's Series 40 platform [384], which is not Symbian OS-based and runs on the "dumb" phones and many feature phones in the company's product range, also has (limited) MIDlet support.

[12] The original iPhone came out in the US in June 2007, but it took until November '07 before it was officially put on sale in a few other countries. Apple's smartphone only became widely available when the iPhone 3G, the 2[nd] generation model, was released simultaneously across 70 countries in July '08 [586].

[13] Ironically these figures were (re)circulated by Stephen Elop, Nokia's recently appointed CEO, in a public memo to his employees [561], in which he added: « *They changed the game, and today, Apple owns the high-end range* ».

[14] iOS [30] is the operating system which runs on the iPhone and on Apple's iPod touch and iPad products. Apple does not license the platform to other vendors.

[15] Strictly speaking this might change soon as Google is in the process of acquiring Motorola Mobility [220], one of the main manufacturers of Android devices.

All have touchscreens and the more expensive models match or eclipse Apple's offerings in terms of hardware specifications. Google also launched an accompanying app store called Google Play (formerly Android Market) [219]. While sales of Android phones remained modest in 2009, they grew with an explosive 888.8 % in the course of 2010 [201].

In the meantime, once dominant Nokia has struggled to keep up. After having taken full ownership of Symbian Ltd. in late 2008, Nokia merged their S60 layer [591] with Symbian OS to form a complete mobile device platform simply called *Symbian* [519, 592]. While Nokia, and the Symbian Foundation it had set up, continued to invest in the platform, Symbian failed to deliver the features and user experience the more exigent customers had come to expect [561]. This has caused Nokia's sales to decline, especially in the upper regions of the smartphone market. Still, the company managed to limit the damage by focusing on the lower end and by maintaining a strong presence in important developing markets like India and China[16]. Meanwhile other vendors have started to flee the Symbian camp in favour of Android and/or proprietary platforms[17]. Eventually even Nokia realised that Symbian is too outdated to compete at the high-end of the market. In February of 2011 the company announced [381] it has selected Microsoft's Windows Phone [346] as its primary platform for future smartphones, although it plans to continue to sell devices with updated versions of Symbian until at least 2013.

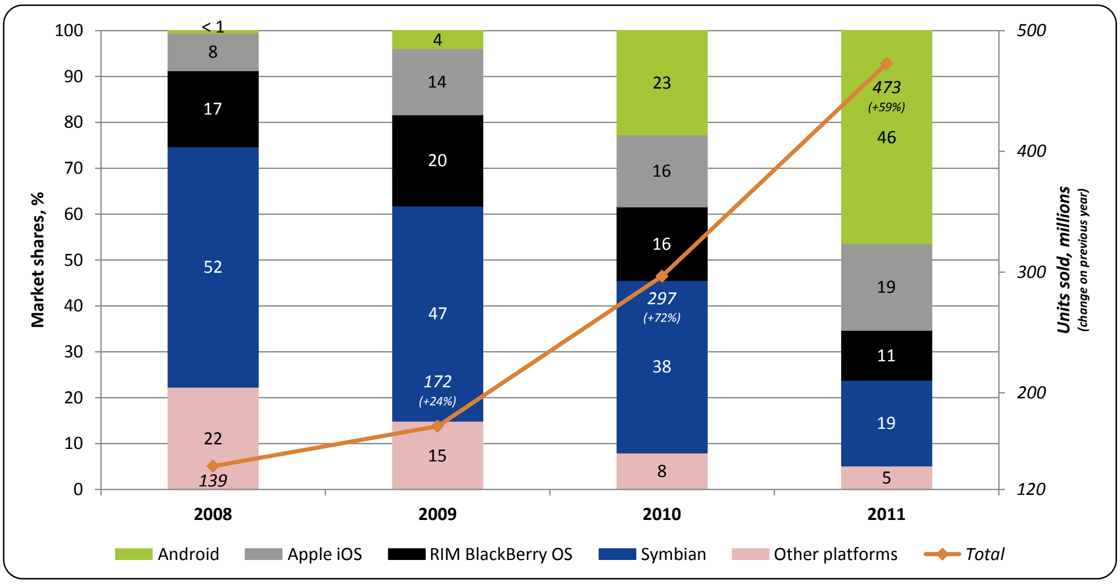


Chart E.1: Evolution of the global smartphone market from 2008 to 2011

Chart E.1 illustrates how these trends have affected the global market for smartphones across all price ranges. We based this chart on end-user sales figures reported by market research firm Gartner [200–205]. The dominance of Symbian has been clearly declining,

[16] In May 2010, Chris Jones, VP of market research firm Canalys, put it this way [77]: « *Aggressive pricing has enabled Nokia to deliver smartphones that appeal to a broader consumer audience* ».

[17] While there are still Symbian-powered devices from other vendors on sale today, since late 2010 only Nokia has announced new Symbian-based products.

but the troubled platform managed to stay in the lead up to 2010, when it ran on 37.6 % of sold devices, down from 46.9 % in 2009. In 2010 sales of Android-running smartphones soared beyond those of Apple's iPhones and RIM's BlackBerries, putting the platform in second position with a market share of 22.7 %, up from just 3.9 % in 2009. Meanwhile, Apple's share grew modestly[18] from 14.4 to 15.7 % and RIM's dropped from 19.9 to 16 %. The rise of Android has continued in 2011, according to Gartner's quarterly assessments, Google's offering has outsold all competing smartphone platforms since the beginning of the year and in Q3 and Q4 it even held slightly more than half of the market [204, 205].

Since 2008, Java ME CLDC/MIDP has lost its momentum as a middleware platform for smartphones. The platform itself has seen little or no innovation[19] and most vendors have stopped investing in it. Developers have also been deserting the platform [554, 555], typically in favour of iOS or Android, neither of which can run MIDlets and which offer superior application development frameworks. Still, even now huge numbers of MIDlet-supporting devices continue to be sold, although many are not considered smartphones by today's standards. Thanks to this evolution, MIDlets that required a high-end smartphone back in 2008 – such as NoiseTube Mobile for Java ME (see chapter 6) – can now run on cheap GPS-equipped feature phones[20].

E.3 The Java ME CLDC/MIDP platform

Java ME [513], or the *Java Platform, Micro Edition* in full, is a middleware platform, based on the Java programming language [515], for the development of applications for mobile devices – primarily mobile phones and personal digital assistents (PDAs) – and other embedded systems (e.g. TV set-top boxes).

In the late 1990s, Sun Microsystems[21] (hereafter called "Sun") aimed to extend their popular Java ecosystem towards the market of handheld devices. This ambition could not be realised with Java SE [514], their desktop software platform, because it was too resource demanding, especially in terms of memory footprint, let alone with their enterprise platform Java EE [512]. Therefore Sun introduced a third Java platform, tailored to the needs and limitations of this class of devices. J2ME[22], as Java ME was initially

[18] It is important to realise that market shares only tell half the story because the overall smartphone market grew by 72.1 % in 2010 (see chart). For example, this means that, while its market share barely grew, in absolute numbers Apple still sold almost twice as many iPhones in 2010 than it did in 2009.

[19] Or it never appeared in commercially available products, such as MIDP v3.0 (see section E.3.2).

[20] Such as the Nokia 5230 [386] we used in our validation experiments (see chapter 7).

[21] Since an acquisition in 2010 Sun Microsystems is a subsidiary of Oracle Corporation.

[22] J2ME stood for *Java 2 Platform, Micro Edition*. Likewise Sun's Java offerings for the desktop and enterprise markets used to be called respectively *Java 2 Platform, Standard Edition* (J2SE) and *Java 2 Platform, Enterprise Edition* (J2EE). In 2006 the "2" was dropped from all platform names and "Java" is no longer abbreviated to "J".

called and is still frequently referred to, was first announced and demonstrated at the 1999 JavaOne conference [340]. Figure E.1 shows an overview of the Java ME platform and the family of other Java platforms.

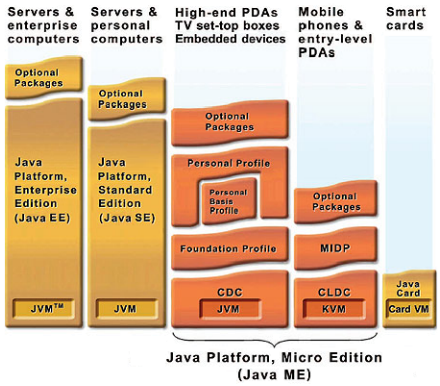


Figure E.1: Overview of Java ME and the other Java platforms offered by Sun/Oracle

To enable deployment across devices with diverse features and limitations, Java ME has a modular architecture based on three layers [507]: at the lowest level a *configuration* specifies the capabilities of a Java virtual machine [321] and provides a basic set of class libraries; on top of that, a *profile* provides a set of APIs that support a specific range of devices; finally, *optional packages* can add technology-specific APIs. By combining a configuration with a compatible profile (possibility extended with optional packages) different varieties of the Java ME platform have been defined. The *Java ME CLDC/MIDP platform*, which is primarily aimed at mobile phones, is arguably the most common variety.

Contrary to Java SE and EE, Sun chose not to market Java ME as a ready-to-use product to device manufacturers, let alone end-users. Instead, they collaborated with manufacturers – in committees working under the *Java Community Process*[23] – to draw up an ensemble of specifications, each of which typically covers one of Java ME's configurations, profiles or optional packages. Usually no reference implementation is provided, as the task of implementing the specifications is left to manufacturers. In doing so, a manufacturer can create a Java ME-compliant *runtime environment*[24] and integrate it into products to let them run Java ME applications. While the specifications themselves can be freely consulted by anyone, the implementation and distribution of a runtime environment may require licensing patented technologies and possibly trademarks from Sun/Oracle. In line with the « *Write once, run anywhere* » slogan, with which Sun used to promote the Java ecosystem, Java ME applications are in principle platform-independent, in the sense that they can run on any operating system (OS) as long as there is a compliant runtime environment available for that OS.

[23] Established by Sun in 1998, the Java Community Process (JCP) [518] allows interested parties to get involved in the definition of future versions and features of the Java platform(s).

[24] Examples are Nokia's *Java Runtime for Symbian* [380] and the *Sony Ericsson Java Platform* [487], both of which implement the Java ME CLDC/MIDP platform for use on smartphones.

To support application developers Sun/Oracle offers the *Java ME Software Development Kit* (SDK) [511] which can be downloaded for free. It includes documentation and all the necessary libraries to compile Java ME applications as well as a device emulator for testing purposes. Typically manufacturers of CLDC/MIDP devices (e.g. Nokia and Sony Ericsson) also offer their own SDK, containing documentation specific to their products and emulators which are technically closer to the actual devices.

In what follows we will discuss each of Java ME's layers. We will focus especially on the specifications relevant for mobile phone applications.

E.3.1 Configurations

As shown in figure E.1 two *configurations* have been defined for the Java ME platform: the *Connected Limited Device Configuration* (CLDC) [509] and the *Connected Device Configuration* (CDC) [508]. With the CLDC specification Sun was aiming at small, resource-constrained devices like mobile phones. The more capable CDC was aimed at larger devices with more capacity, like high-end smartphones, embedded devices, and TV set-top boxes. The class libraries provided by both configurations are subsets of standard Java SE libraries, complemented with additional classes specific to Java ME and the configuration in question. Because it is aimed at more powerful devices CDC contains a larger subset of Java SE libraries than CLDC does and it specifies a virtual machine that is closer (or equal) to the one that powers Java SE [510].

Despite Sun's efforts, only the CLDC-based variety of Java ME enjoyed wide popularity as an application platform for mobile- or smartphones. With the *Mobile Information Device Profile* (MIDP) layered on top of it (see below), CLDC was adopted by many mobile phone manufacturers; most notably Nokia, Motorola, RIM, Sony Ericsson and Samsung. Even as devices became more powerful, CDC was largely left aside, most likely due to existing investments in CLDC implementations and its popularity among application developers. The CLDC/MIDP combination became the dominant application platform for smartphones and feature phones, until the arrival of Apple's iPhone and Google's Android platform thoroughly changed the smartphone market (see section E.2).

The latest version of CLDC is v1.1.1, which is minor update of the earlier v1.1 [284]. In comparison with the first version – i.e. v1.0 [279] – v1.1 added many important features, such as floating point support. CLDC v1.1 and v1.1.1 require applications to be written in the Java programming language as defined by the second edition of the Java Language Specification [228]. Language features introduced in the third edition [229] are not supported. The class libraries defined by CLDC can be divided into two categories.

The first category is a subset[25] of the standard Java SE libraries, containing fundamental classes which are closely tied to the Java language and runtime environment (e.g. data type, collection and exception classes). Since CLDC v1.1 this subset is derived from J2SE v1.3.1. The other category is formed by the *Generic Connection Framework*, an abstract I/O and networking library which is specific to CLDC.

E.3.2 Profiles

As mentioned above, Java ME configurations are intended to be complemented with a *profile*. To complement CLDC on so-called *Mobile Information Devices* (MIDs), which are typically mobile phones or PDAs, Sun and its partners defined the *Mobile Information Device Profile* (MIDP) [517].

Apart from an additional trio of Java SE-derived classes[26], the MIDP specifies [280] a set of class libraries that are specific to the profile. Most importantly it contains a library which defines the nature of applications, called *MIDlets*[27], and their interaction with the runtime environment. Furthermore MIDP contains a user interface (UI) toolkit, called *LCDUI*, which is aimed at small LCD screens. Another library extends CLDC's Generic Connection Framework to provide, among other things, APIs for HTTP communication. Last but not least there is a library that deals with persistent data storage (though no file system access is provided). In version 2.0 [282] of MIDP all of this was complemented with APIs that deal with security certificates, gaming features and a *Media API* library which provides support for audio playback. The MIDP specification has since received a minor update to v2.1 [282] and later a major overhaul to v3.0 [288]. However, to the best of our knowledge, there are no commercially available MIDP v3.0 devices yet[28].

To build a MIDlet, one must create a subclass of `javax.microedition.midlet.-MIDlet`. That class then serves as the entry point of the MIDlet, which typically spans multiple additional classes. The whole of one or more MIDlets is packaged for distribution in a self-contained bundle called a *MIDlet suite*. This is a `.jar` archive file which contains the compiled classes (`.class` files) of the MIDlets and the libraries they may require

[25] Regarding this subset, the CLDC specification [284] follows the general rules for Java ME configurations. These stipulate that each provided class or interface that has the same name and package name as one from Java SE must be identical to or a subset of the corresponding Java SE class or interface. Furthermore, its semantics and those of subset-retained public or protected methods and fields cannot be changed and no public or protected methods or fields can be added.

[26] Strictly speaking, the classes in question (`java.lang.IllegalStateException`, `java.util.Timer` and `java.util.TimerTask`) are only in MIDP up to v2.1 [282]. With the release of the CLDC v1.1.1 [284] and MIDP v3.0 [288] specifications they were moved "down" from MIDP to CLDC. Hence, MIDP v3.0 no longer includes any Java SE-derived classes.

[27] The word "MIDlet" is a portmanteau of *MID* and *applet*, which is a term for small software applications.

[28] Possibly that will never change because the spec has been ready since 2009 and even Motorola, which led its development, never announced MIDP v3.0 devices and has since abandoned Java ME for Android.

(see below), additional resources (e.g. image files) and a manifest file. The `.jar` file is usually accompanied by a `.jad` descriptor file. Nowadays, MIDlets are often distributed through app stores, such as the Nokia Store [382] among others.

E.3.3 Optional packages & vendor APIs

As mentioned above, Java ME foresees *optional packages* to extend configurations and profiles with APIs for specific technologies. As the name suggests, the implementation of optional package specifications is non-compulsory, in the sense that device manufacturers can decide whether or not to support particular packages independent of the decision to implement a particular configuration and profile. This can cause headaches for developers and users alike because the fact that a device supports CLDC and MIDP does not always guarantee that it supports all the optional packages a given MIDlet relies on[29].

Like CLDC and MIDP, optional packages are developed as a *Java Specification Request* (JSR) under the JCP. Hence they are commonly referred to as "JSR-*x*", in which *x* is the number their specification was given under the JCP.

Some commonly used examples of optional packages are:

- **JSR-135: *Mobile Media API* (MMAPI)** [283]
  This package extends the multimedia functionalities exposed by the MIDP v2.0 Media API. In fact, the MIDP v2.0 Media API is defined as a subset of the MMAPI.

- **JSR-179: *Location API*** [285]
  This package adds an API that exposes information about the present physical location of the device, provided that the user authorises this. While devices that implement JSR-179 commonly obtain this information from a GPS receiver (either an integrated or an external one), the API itself is agnostic with respect to the used positioning technology. Instead programmers can specify a set of criteria (in terms of accuracy, timeliness, cost, etc.) for the selection of a `LocationProvider`.

- **JSR-75: *FileConnection Optional Package* (FCOP)** [281]
  This package specifies an API for file system access. Provided the user allows it, this enables MIDlets to read and write files and to create, rename and delete files or directories on the device's internal memory or on removable memory cards.

- **JSR-75: *PIM Optional Package* (PIMOP)** [281]
  Apart from the FCOP, JSR-75 (called *PDA Optional Packages*) also defines this

[29] We should note that there have been efforts to counter such fragmentation of the platform by means of "umbrella" specifications [286, 287] covering CLDC, MIDP and fixed sets of optional packages.

separate package which deals with personal information management (PIM) features such as contacts and schedules.

Like the other parts of the Java ME platform the specifications of optional packages can be updated from time to time. Developers should thus be careful and not only specify which optional packages their MIDlets require but also which versions thereof.

Some vendors – notably Nokia and RIM – include supplementary, independently developed class libraries in the Java ME runtime environment they ship on their devices. Such *vendor APIs* expose functionalities which are specific to the vendor's products (devices and/or services). In the hierarchy of Java ME layers vendor APIs are on the same level as the optional packages. But unlike those, vendor APIs are not part of the ensemble of Java ME specifications (developed under the JCP). An example of a vendor API is:

- ***Nokia UI API*** [383]
  Nokia introduced this MIDP extension in order to expose specific features of their devices. Being a vendor API, it was not standardised as a JSR, but nevertheless it is also supported on some Sony Ericsson phones [487].

E.3.4 Permissions

The usage by MIDlets of many features exposed by optional packages – but also of some functionalities provided at the level of MIDP, such as Internet access – is restricted through a system of explicit permissions, controlled by the user of the device.

The MIDP permissions system works on a per-application basis and is *selective* in the sense that users have the freedom to grant some permissions while declining others requested by the same application (e.g. "I allow MIDlet X to access the Internet but it should not know where I am."). The permissions which a MIDlet requires (along with optional ones) have to be declared in the manifest of the `.jar` file it is distributed in (and also in the accompanying `.jad` file). Unfortunately few devices[30] use this information to let the user grant or decline permissions upon the installation of a MIDlet. Instead, users can take the initiative and find their way through the settings panel of the device to configure the permissions of installed MIDlets, although none of the devices we have seen indicate which permissions are essential or optional to particular MIDlets. Alternatively users can wait until they are prompted to grant (or decline) permissions during the execution of the application, which happens every time an application tries to use a restricted feature for which it has not been granted (permanent) permission beforehand.

[30] As far as we know, only recent Nokia smartphones (running v2.1 or later of the Java Runtime for Symbian [380]) and (some) Sony Ericsson devices offer to help the user with configuring permissions as part of the MIDlet installation process.

The selective nature of the permission system, the fact that previously granted permissions can be revoked and that permission requests can happen at runtime all make it necessary for developers to implement fall-back mechanisms to avoid that their MIDlet crashes in case a permission is declined. If the declined permission is essential for the application to do anything meaningful this may mean an error message should be shown, possibly followed by a graceful exit of the program.

As an additional protection against malicious intent MIDP's permission system is coupled with a digital signing process. In short, the set of restricted features a third-party (i.e. not preinstalled by the manufacturer or a network operator) MIDlet can access, and whether or not it needs the user's permission, depends on whether it is *trusted* or *untrusted*. For a MIDlet be trusted the MIDlet suite it is distributed in needs to be digitally signed with a certificate that identifies the creator of the software. Such certificates have to be bought from a *Certificate Authority* (CA) and have a limited validity (typically 1 to 3 years)[31].

E.3.5 Additional libraries

Apart from relying on the different layers of the Java ME CLDC/MIDP platform, MIDlet developers can of course use additional third-party libraries, provided these are compatible with CLDC. Because MIDlet suites must be self-contained, all required libraries have to be included (in compiled form) in the same `.jar` package the MIDlet(s) are distributed in.

A popular example of a library used in MIDlets is the *Lightweight UI Toolkit* (LWUIT) [516]. LWUIT is developed by Sun/Oracle as an open source project[32] and constitutes a welcome alternative for MIDP's built-in acrLCDUI toolkit, which has many limitations (e.g. the lack of support for touchscreens). In fact, one of Sun's motivation to develop LWUIT was precisely to address the limitations of LCDUI and thus provide a more advanced widget toolkit for CLDC/MIDP [516].

E.3.6 Summary

Figure E.2 summarises the stack of components typical MIDlets depend on. As a concrete example, the diagram mentions the names and/or versions of all components in the case of *NoiseTube Mobile for Java ME* (see chapter 6) running on the Symbian-based [519] Nokia 5230 [386]. The orange parts in the diagram constitute the layers of the Java

[31] Before a certificate is delivered the issuing CA does background checks to confirm the identity of the buyer. We have bought our certificate(s) from VeriSign, one of the major companies in this business, for about US$ 500 per year.

[32] LWUIT is released under version 2 of the GNU General Public License (GPL) [194] with the "classpath exception" [584], enabling linking of the library to non-GPL licenced programs.

ME middleware as it is implemented on this particular device by the *Java Runtime for Symbian* [380]. The green parts correspond to the MIDlets which can be installed on the phone and the additional libraries those might depend on and include. For completeness the diagram also shows native applications (with an example) which run directly on the operating system, alongside the Java ME runtime environment.

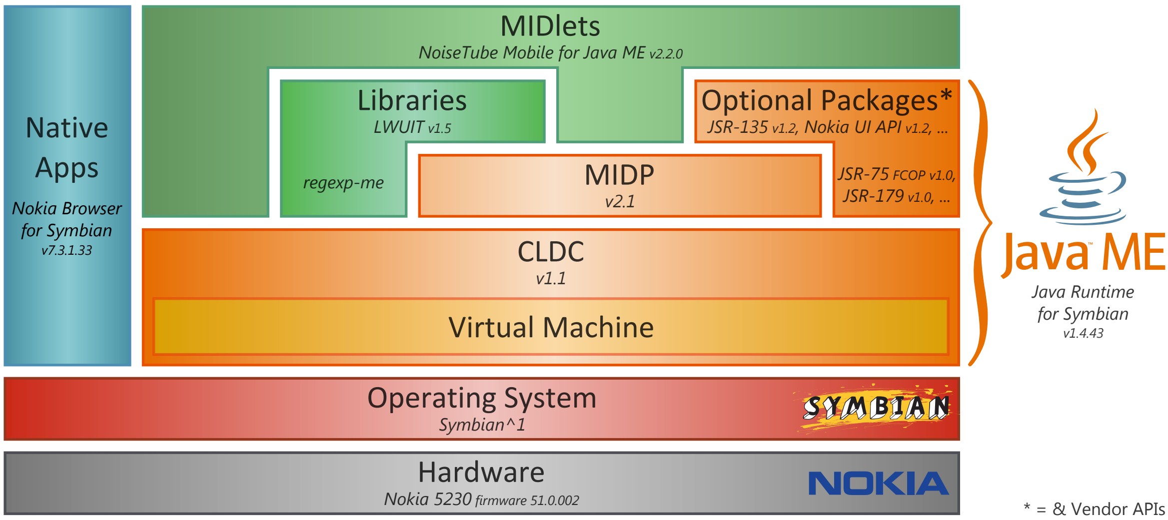


Figure E.2: Component architecture of a typical Java ME CLDC/MIDP device running MIDlets alongside native applications

E.4 The Android platform

Android [226] is a complete, open source software stack for mobile devices. Development of Android started in 2003 by Android Inc. which was acquired by Google in 2005. Since then Android has served as the centrepiece of Google's strategy to bring its portfolio of Web applications and services to users of mobile devices anywhere. Google leads the development and promotion of Android in collaboration with the Open Handset Alliance (OHA) [397], a Google-founded partnership of interested parties[33].

Two noteworthy differences between Android and Java ME come to mind immediately. On the one hand there is a difference in scope: while Java ME only offers a middleware solution for mobile apps, Android also includes an underlying operating system – based on the Linux kernel [531] – and a suite of core apps [224]. On the other there is a difference in the nature of the product: while Java ME was put forward by Sun as an ensemble of specifications to be implemented by device manufacturers, Google and the OHA provide a ready-to-use implementation of Android to manufacturers. However, like Sun/Oracle,

[33] Chiefly mobile phone manufacturers, network operators, semiconductor and software companies.

Google does leave the design and manufacturing of actual devices to other companies. In contrast, Apple's iOS platform [30], Android's main competitor in today's market, is exclusively shipped on products from Apple itself.

Similarly to iOS, Android is aimed at both smartphones and tablets. The first publicly available beta version of Android was released in 2007 and the platform reached v1.0 in September 2008, in time for the first commercially available Android device, the HTC Dream, which was released in October 2008 [575]. Since then, over 200 different Android devices have been released or announced [578]. Currently the main vendors of Android-powered smartphones are Samsung, HTC, Sony Ericsson and Motorola[34]. The current Android version for both smartphones and tablets is v4.0.x, codenamed *Ice Cream Sandwich*, which was released in October 2011. Before this version, there were separate editions for smartphones – v1.0 to v2.3.x – and tablets – v3.0.x to v3.2 [575]. For app development purposes a parallel versioning scheme of so-called *API levels* [223] is used. Currently, the most recent API level is 15, which corresponds to v4.0.3.

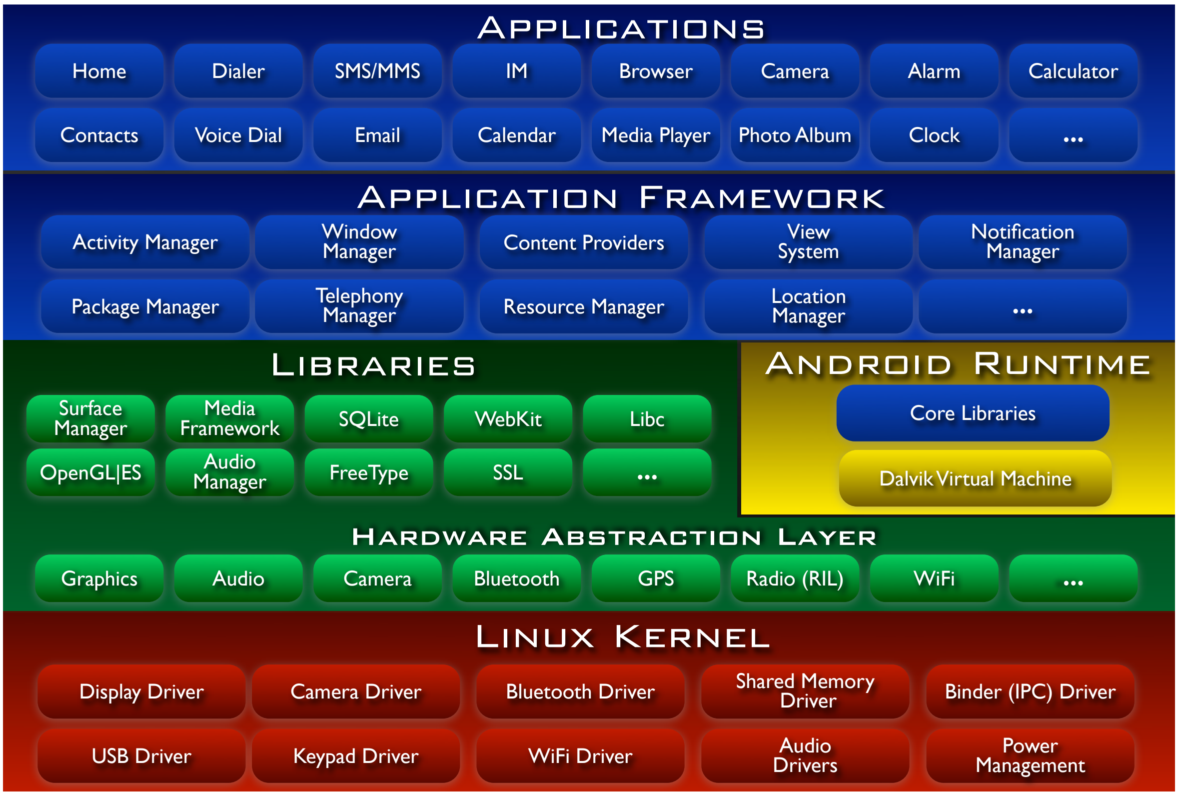


Figure E.3: Overview of the Android software stack architecture

Figure E.3 shows an overview of the architecture of the Android software stack. At the bottom, the Linux kernel forms the core of the operating system. Strictly speaking this is

[34] According to ABI Research, a market research firm, these vendors respectively accounted for 34, 23 11 and 9 % of the total 47 million Android smartphones shipped in Q2 2011 [3].

not the official Linux kernel as it incorporates a number of Android-specific enhancements, primarily to deal with inter-process communication (IPC) and power management (an important concern on mobile devices). On top of the kernel sits a layer of C/C++ system libraries which are used by different parts of the Android platform and form an integral part of the operating system. This includes the *hardware abstraction layer* which provides the interface between the kernel drivers for various hardware components and the rest of the system. Quite a few of the system libraries are derived from other open source projects (e.g. WebKit, SQLite, FreeType, etc.). The capabilities of the system libraries are exposed to app developers through the *Android Application Framework*.

Generally speaking, Android apps are developed in the Java programming language [515] using the *Android Software Development Kit* [227]. The SDK includes documentation, libraries for compilation of apps and a device emulator for testing purposes. Despite employing Java as the main app development language, Android does not align with Java SE nor Java ME. Instead, the Android open source project has developed its own virtual machine, called *Dalvik*, and has assembled its own set of class libraries. Android apps rely on two layers of class libraries: the *Core Libraries*, which together with the Dalvik virtual machine form the *Android Runtime*, and the libraries provided by the Android Application Framework (see below). There are some alternatives to Java for programming Android apps and we will briefly discuss those as well.

Android includes a suite of core apps which are an integral part of the system. These apps, examples of which are shown in the top layer in figure E.3, provide the usual functionalities expected of a smartphone or tablet device (e.g. home screens, making phone calls, SMS/MMS messaging, contact management, web browsing, etc.). Typically these come pre-installed on Android devices, although manufacturers have the possibility to change the "look & feel" or replace some of them with their own alternatives, in an effort to differentiate their Android-powered products from those of other manufacturers. There is little or no difference between such pre-installed apps and the ones developed by third-parties (downloaded and installed by users themselves): all apps are executed by the Android Runtime and rely on the Android Application Framework. Android apps are packaged for distribution as an `.apk` file which contains the compiled app code (see below), additional resources and a descriptor. The distribution and sales of most apps happens through the Google Play [219] service, although there are a number of other, competing services. Users can also be download and install `.apk` files from any website, but then the user must first disable a safety setting on the device which by default blocks the installation of apps downloaded from "untrusted" sources.

E.4.1 Android Runtime

The Android Runtime consists of the Dalvik virtual machine (VM) and the Core Libraries. This combination of a virtual machine and a set of class libraries makes the Android Runtime similar to Java ME's *configuration* layer (see section E.3.1). Therefore we will discuss how the Android Runtime compares to CLDC [284, 509], the configuration found on Java ME-supporting mobile phones.

The Dalvik VM [52] was developed from scratch for the Android platform. It is optimized for mobile devices and relies on the Linux kernel and system libraries for underlying functionality such as threading and low-level memory management. Dalvik is used to execute (or at least bootstrap) all Android apps as well as the supporting Application Framework and Core Libraries (i.e. all the "Java parts", shown in blue in figure E.3). Dalvik was designed to allow mobile devices to efficiently run multiple concurrent VM instances. This is necessary because Android supports multitasking and every app runs in its own process, with its own VM instance. Similarly to HotSpot [510], the primary VM powering the Java SE and EE platforms, Dalvik increases performance though *Just-In-Time* (JIT) compilation as of Android v2.2, codenamed *Froyo*. However, Dalvik differs in a number of ways from HotSpot and other Java VMs (which follow Sun's specifications [321]). Unlike Java VMs which are stack machines, Dalvik has a register-based architecture. Instead of Java bytecode, Dalvik uses its own, more compact, instruction set. Consequently the Java class file format (with `.class` extension) is not used either. Instead Dalvik executes files in the Dalvik Executable format (`.dex` extension) which is optimized for minimal memory footprint. Yet conveniently, developers can continue using existing Java compilers because after compiling an app's Java source code to a set of `.class` files (possibly packaged in a `.jar` archive) these can be transformed into a single `.dex` file using the "dx" tool, which is included in the Android SDK.

The Android Core Libraries (ACL) are built around a subset of the *Apache Harmony* Java implementation, complemented with a few Dalvik-specific classes and a handful of libraries derived from other open source projects (e.g. JUnit and SAX). Led by the Apache Software Foundation, Apache Harmony is an open source project [24] which aims to create a "clean-room" implementation[35] of Java (SE), licensed under the Apache License v2.0 [28]. Harmony achieves close to 100 % API completeness with respect to both v5.0 and v6.0 of Java SE [23]. Like the Java SE subset found in CLDC (see section E.3.1), the Harmony subset in ACL constitutes a set of fundamental classes closely tied to the Java language (e.g. data type, collection and exception classes). The fact that these are derived from the Harmony project is entirely transparent to app developers. All classes (and interfaces) included in the subset carry the same name, are grouped in the same

[35] Wikipedia defines [576] a *clean room design* (also known as the *Chinese wall technique*) as the method of copying a design by reverse engineering and then recreating it without infringing any of the copyrights and trade secrets associated with the original design.

package hierarchy and expose a virtually identical set of public and protected methods and fields as the "originals" do in the Java SE version the Harmony implementation is based on. This means that, even though it is – strictly speaking – not Java SE-compatible, Android quickly makes Java programmers feel "at home". What's more, existing Java code which only relies on classes within the subset typically works on Android without or with only minor modifications. Given the huge number of developers with some level of Java experience[36] this is a one of the main strengths of the Android platform[37].

The Harmony subset in ACL does not include Java SE's AWT and Swing libraries. Instead, Android has its own UI toolkit which is part of the Application Framework (see below). Similarly, CLDC does not include AWT and Swing either and instead a UI toolkit is provided by MIDP. Nevertheless, compared to CLDC, ACL contains a much bigger subset of Java SE[38], albeit in "Harmony form", which, as explained above, hardly matters in practise. Besides being bigger the Harmony subset also completely encompasses CLDC's Java SE subset, meaning that all of CLDC's Java SE-derived classes/interfaces are also present in ACL. The internals of Java SE-derived classes/interfaces can also differ between CLDC and ACL. While in CLDC some methods and fields have been dropped [284], Harmony's implementation of Java SE libraries is virtually complete and no methods or fields were left out in the subset included in ACL. Finally there is a difference in the Java SE version the subsets are derived from. While even the latest release of CLDC (v1.1.1) still derives from J2SE v1.3.1 (dating back to 2001), ACL's subset is derived from Harmony's implementation of J2SE v5.0 until Android v2.2 *Froyo* and from Harmony's Java SE v6.0 branch since Android v2.3 *Gingerbread*. All of this means that, in terms of fundamental classes, Android is much closer to Java SE than Sun/Oracle's own Java ME platform. The situation of the Java SE subsets in CLDC and ACL – which is the only place where they overlap – is summarised in figures E.4 and E.5.

Another noteworthy difference between the Android Runtime and CLDC is the support for features of the Java language itself. While CLDC v1.1 and v1.1.1 require programs to be written in the Java language as defined by the second edition [228] of the Java Language Specification (JLS), released in 2000, the Java code of Android apps can be written in compliance with the third edition [229], released in 2005 and still the latest formal JLS. This means Android developers can use powerful language features such as generics, annotations, `foreach` loops and `enum` types. Already in 2004 – with the release of J2SE v5.0 – Java SE received support for the these *third edition* features. Now, almost eight years on, it seems unlikely that they will ever be supported on CLDC.

---

[36] Quoting a 2009 survey conducted by Evans Data Corporation, a market research firm, Oracle states [452] there are globally over 9 million Java programmers, more than for any other language.

[37] In contrast, Apple's iOS platform uses the less popular Objective-C as its main app development language, although this has hardly hindered its success [32].

[38] The collection classes are one area where the difference is clear: while CLDC only includes the `Vector`, `Stack` and `Hashtable` classes, ACL contains the complete suite of Java SE collection classes.

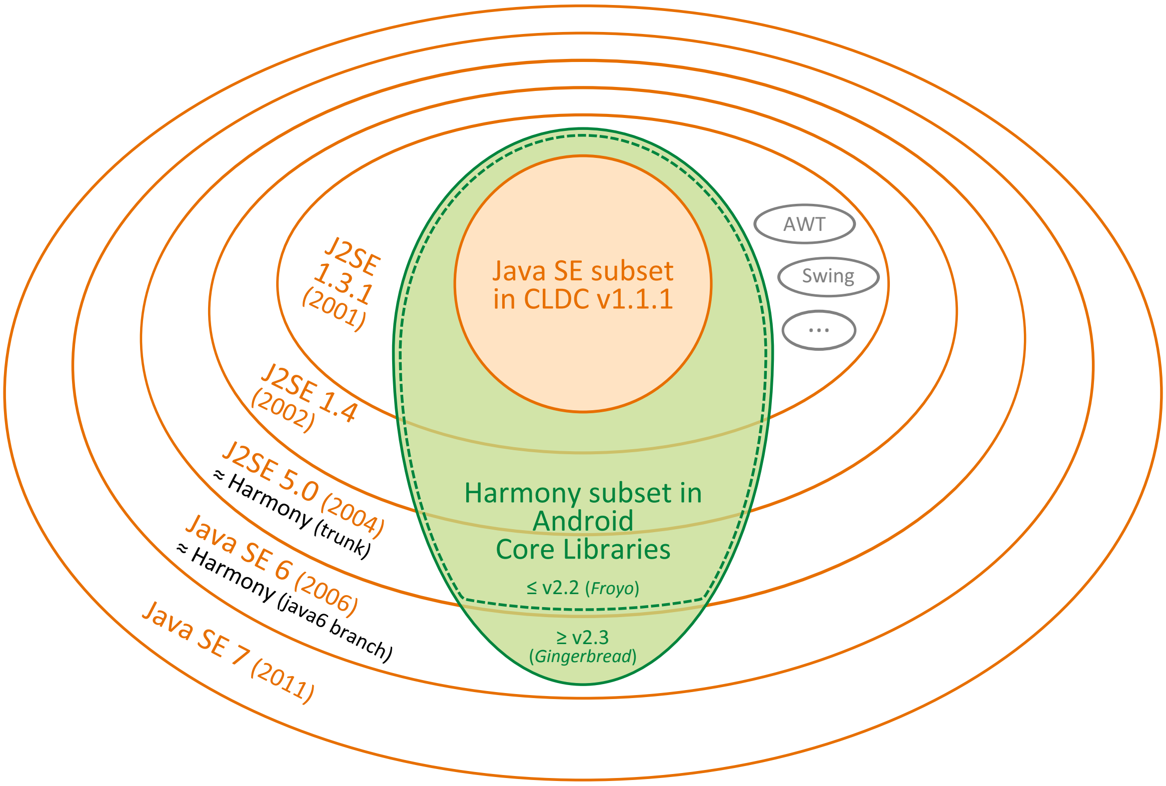


Figure E.4: Comparison of the subsets of Java SE (formerly *J2SE*) class libraries found in CLDC and the Android Core Libraries

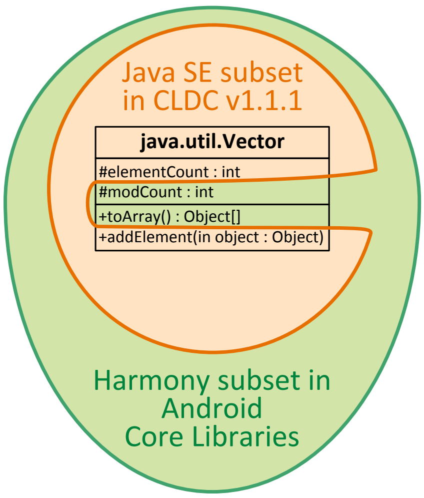


Figure E.5: Detail of the overview in figure E.4, showing an example of a Java SE-derived class from which some fields and methods (an example of each is shown) were not retained in CLDC's Java SE subset, while being supported in the Android Core Libraries

While the choice to develop the Dalvik VM was at least partially motivated by technical concerns, the use of code from the Harmony project arguably has more to do with commercial and legal strategies [471]. It is what enabled Google and the OHA to benefit from the popularity of Java without having to license technology from Sun/Oracle[39], being restricted by the limitations of Java ME, having to deal with the Java Community Process or resort to other open source Java implementations with incompatible licenses such as *IcedTea* [445, 483].

E.4.2 Android Application Framework

The Android Application Framework (AAF) provides the second class library layer for Android apps to rely on. Through the AAF app developers have full access to the same framework APIs used by the core apps (see above). The framework defines with the nature of Android apps and their interaction with the platform environment. Furthermore it provides a UI toolkit and facilitates interaction with a wide variety of device features (telephony, contacts, media playback and recording, data storage, notifications, etc.), network interfaces, Internet services, integrated sensors, etc.

In the previous section we argued that the Android Runtime was similar to Java ME's *configuration* layer and we discussed how it compares to CLDC. Similarly there are good reasons to say the AAF roughly corresponds to Java ME's *profile* layer (see section E.3.2). However, in comparison to MIDP [282, 517], the Java ME profile for mobile phones, the AAF is a much more modern and comprehensive mobile application framework. Many features requiring *optional packages* (which may not be universally supported) on MIDP devices are included in the baseline Android platform thanks to APIs exposed by the AAF. Examples are media recording, sensor readouts, Bluetooth connectivity, access to location information, etc. We should also note that the AAF class libraries are incompatible with MIDP and its optional packages[40].

Typical Android apps consist of one or more *Activities* and/or *Services*. An activity is created by subclassing the `android.app.Activity` class and it represents a single, focused thing that the user can do. The `Activity` class takes care of creating a window on the screen in which a user interface can be placed. A service on the other hand is an app component (a subclass from `android.app.Service`) which is used to perform long-running operations while not interacting with the users (or only through sporadic notifications) or to supply functionalities for other apps to (re)use.

The AAF's UI toolkit provides an extensible set of *View* classes that cover a variety of layouts and widgets. The toolkit is not based on, nor compatible with, other toolkits

[39] This may not it remain without consequences because in August 2010 Oracle started a (still on-going) lawsuit against Google, claiming that "[...] *in developing Android* [and Dalvik], *Google knowingly, directly and repeatedly infringed Oracle's Java-related intellectual property*" [369].

[40] However, there the are some striking similarities. For example, the AAF classes that deal with locations and positioning technology (e.g. GPS) seem inspired by the JSR-179 Location API [285].

such as MIDP's LCDUI, LWUIT or Java SE's AWT and Swing. The UI of an Android app can be defined entirely in Java code or using an XML-based description language, or a combination of both. The XML form can also be edited in a WYSIWYG-fashion[41] using the *Visual Layout Editor* which is part of the official *Android Development Tools* plugin [222] for the Eclipse IDE [141]. At runtime UIs are brought to life by the *View System* component, shown on figure E.3, which acts behind the scenes to manage the UI elements which are in view and to trigger events as users interact with the UI.

Apart from the View System, the backbone of the AAF is formed by a set of *Manager* and *Provider* classes, also called *runtime services*. Figure E.3 shows some examples of these. Not to be confused with the app level services mentioned above, runtime services cannot be directly instantiated by apps. Instead, instances can be requested through a factory method [198] of the `android.content.Context` class. The main purpose of the runtime services is to expose capabilities of underlying system libraries which they invoke using *Java Native Interface* (JNI) [319] calls. Informative examples of such interactions across the layers of the Android stack are discussed in [56].

The only part of the AAF which was not developed from scratch is the *HttpClient* library. This one was derived from the Apache HttpComponents project [25] and provides a standards based, pure-Java implementation of the HTTP protocols.

E.4.3 Permissions

The use by apps of some AAF functionalities is restricted through a system of permissions. Like the MIDP permission system (discussed in detail in section E.3.4), Android's permission system works on a per-app basis. Like MIDlets, Android app packages (`.apk` files) contain a descriptor which stipulates the permissions the app requires. However there is an important difference between MIDP's and Android's permission system. MIDP has a *selective* permissions system, which means users can choose to grant some of the permissions a MIDlet requests while rejecting others. By contrast, Android's system follows an *all-or-nothing* approach which forces users to grant either all or none of the requested permissions. A related issue is the moment at which the question is asked. On CLDC/MIDP devices it is common that users are not asked to grant permissions until during the execution of an app. On Android the question is asked only once, namely upon installation. During that procedure the user is informed about the permissions an app needs and, as a literal interpretation of all-or-nothing, the app is only effectively installed when he/she agrees with the whole proposition (not agreeing cancels the installation).

From the perspective of the user there are advantages and disadvantages to both approaches. The selective nature of MIDP's system can increase trust among users because

[41] *What you see is what you get.*

it gives them fine-grained control over what apps can and cannot do. This can be valuable in case of apps which provide services users want but do so at the cost of a (perceived) invasion of privacy[42]. On the other hand, an all-or-nothing system such as Android's can be simpler to use (especially for novice users) because there is only a single choice to be made and it should only be made once. It also avoids that users are faced with half-functioning (or even crashing) apps because they did not grant all necessary permissions.

From the perspective of app developers Android's permission system is clearly favourable over MIDP's. The all-or-nothing approach ensures that any running app can use all restricted features it has asked permission for, because otherwise it simply would not be installed, let along running. This relieves programmers from the burden of having to write fall-back routines and helps them to create a consistent user experience.

It should be noted that independently of permission systems developers obviously have the freedom to include a preferences screen within their app to allow users to disable certain behaviours on the basis of privacy or other concerns. Of course this implies that the user trusts the developer to begin with (i.e. that the app respects the user's wishes). When this trust is absent, a selective, revocable permission system such as MIDP's provides users with an additional safeguard against malicious intent. On Android, by contrast, mistrusted apps can only be revoked their permissions by removing them altogether.

Like MIDP's, Android's permission system is also coupled with a signing process. However, fortunately for developers the certificate needed to sign an Android app can be generated by the creator and so does not need to be bought from an independent CA.

E.4.4 External Libraries

To complement the building blocks discussed above, Android devices typically come with a preinstalled[43], vendor-picked selection of supplementary class libraries which app developers can use. These *external libraries* are not shown as a layer in figure E.3 because, as the name suggests, they are external to the Android software stack. Generally they are not developed under the umbrella of the Android project and may not be open source at all. These libraries roughly correspond to the *vendor APIs* – and to a lesser extent the *optional packages* – found on Java ME devices (see section E.3.3).

Some external libraries are developed by device vendors as part of their efforts to customise, replace or add to the set of core Android apps. Others may be created by third

[42] This situation is all too common among mobile apps, especially those that deal with location-based services, social networking or both. However, permission systems are seldom a magical solution here, for example when the provided service can technically only work if the user is indeed willing to give up some private information.

[43] In some cases end-users can also manually install a missing libraries on their device.

parties and licensed by vendors for inclusion on their devices. For instance, Google offers a number of external libraries that provide functionalities – usually tied to one of its web services – which it has chosen to keep out of Android and its open source license. The external libraries from Google follow the versions/API levels of Android itself.

A popular example is Google's *Maps External Library* [218], which enables apps to integrate a `MapView` widget that displays maps or satellite pictures provided by the Google Maps service [216]. Since it is not part of the Android stack itself the library is not necessarily available on all devices. However, as most vendors do indeed license Google's external libraries, it is supported by a large majority of Android devices in use today.

E.4.5 Native code & other Java alternatives

Instead of programming them in Java it is also possible to write (portions of) Android apps in C or C++. In this scenario the *Android Native Development Kit* [225] is used to compile C/C++ source code to *native* machine code which can be executed directly on the device's CPU instead of being interpreted by a VM. Typically only specific parts of an app are written in C/C++ (if it is used at all) – usually for performance reasons and/or to reuse an existing code corpus. This allows the Java parts of such *mixed* apps to still rely on the APIs provided by the AAF (which are not accessible from native code). The execution of mixed apps always starts on the Android Runtime, where the Dalvik VM interprets the compiled Java code, as the native code parts need to be invoked from Java using JNI [319] calls, similarly to how the AAF's runtime services invoke system libraries (see above). Even apps written entirely in C/C++ start life on the Android Runtime as it executes methods of a generated `NativeActivity` object, which bootstrap execution of the native code via JNI calls.

There are a number of free [14, 214, 221, 373] and commercial [606] ways to build (parts of) Android apps using languages other than Java or C/C++, or even in a visual programming environment [214]. Such solutions typically rely on either (partial) cross compilation to Java or C/C++ or interpretation at runtime by an interpreter written in Java or C/C++ (which is distributed with the app or must be installed beforehand). apps written in the AmbientTalk language [14, 548] – developed at the VUB's Software Languages Lab – are executed on Android using the latter approach. Another interesting solution is the PhoneGap system [373] which allows relatively simple mobile apps to be built using only Web technologies (HTML5, CSS and JavaScript) and targeted at up to 7 different platforms at once (including, besides Android, Apple's iOS, RIM's BlackBerry OS and Symbian). To support Android devices PhoneGap uses a combination of cross compilation (to bind JavaScript calls to AAF APIs) and interpretation (by the JavaScript virtual machine of Android's built-in browser app).

E.4.6 Summary

Figure E.6 summarises the stack of components found on a typical Android device. As a concrete example, the diagram mentions the names and/or versions of all components in the case of *NoiseTube Mobile for Android* (see chapter 6) running, alongside a few other apps (some of which may contain native code parts), on a HTC Desire Z [260], a smartphone powered by Android v2.3 *Gingerbread*.

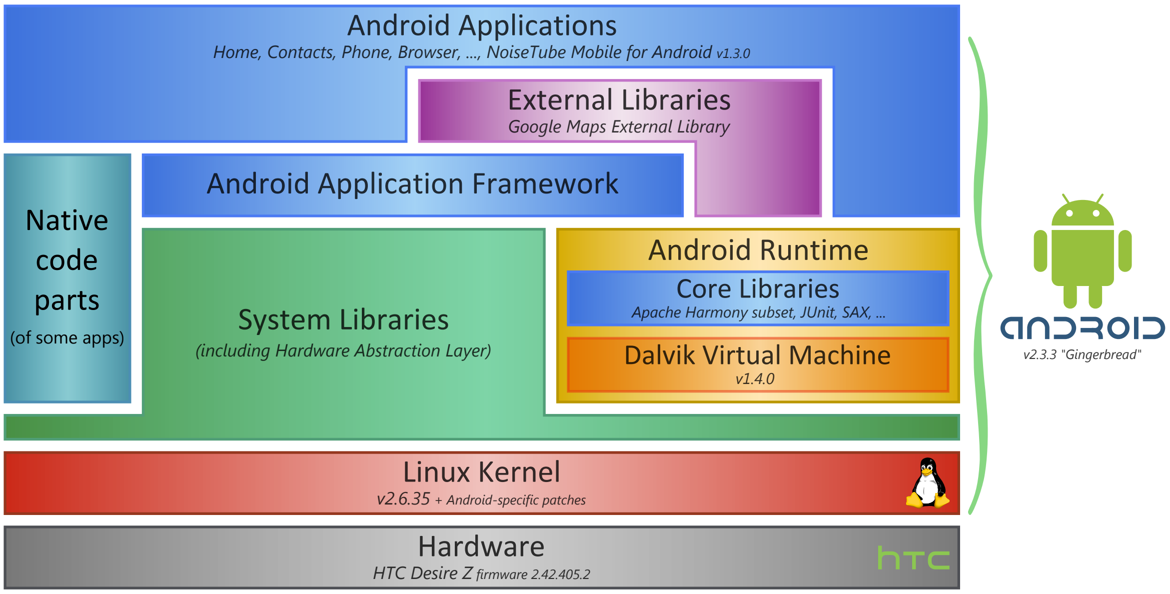


Figure E.6: Component architecture of a typical Android device running multiple apps

It can be interesting to compare this diagram with that in figure E.2, which summarises the components of a typical Java ME CLDC/MIDP device. The difference in scope between Java ME and Android is immediately apparent. At the bottom of the stack we see that Android includes the Linux kernel and a layer of system libraries which together roughly correspond to the operating system layer in figure E.2, which falls outside of the scope of Java ME. At the top of the stack we notice that Android reaches all the way into the applications layer because it includes a suite of core apps (e.g. the browser). The Java ME specifications on the other hand do not describe concrete MIDlets for vendors to include. The middleware layers are more similar: Android's Runtime and its Application Framework (AAF) roughly correspond to respectively the CLDC and MIDP layers in figure E.2. However, there are a number of differences worth remembering. First of all, as discussed in detail in section E.4.1, compared to CLDC the Android Runtime provides a bigger and more recent subset of Java SE classes in its Core Libraries and it supports more Java language features, giving experienced Java programmers a broader set of familiar tools. Second, the AAF provides many functionalities which require optional packages on Java ME phones because they are not covered by the MIDP. On the third middleware layer, the external libraries – which are strictly speaking not part of Android itself – roughly correspond to the vendor APIs found on Java ME devices. Finally we should note that figure E.6 contains no real counterpart to the libraries layer from figure E.2. While it is perfectly possible for developers to use and distribute third-party libraries in their Android apps, we left this layer out of the diagram because we did not use any additional libraries to build NoiseTube Mobile for Android.

E.5 App development guidelines

Based on our experience with NoiseTube Mobile (see chapter 6) we present some guidelines regarding three aspects of mobile app development.

E.5.1 Cross-platform development for Java ME and Android

Sharing or reusing code across the Java ME and Android platforms sounds easier than it is. For one thing, despite the fact that both platforms use Java, their runtime environments support different versions of the language itself. For another, there is only a small set of standard classes that are available on both platforms. We refer to section E.4 for a more detailed discussion of these differences. In the following two sections we outline a solution for the design of cross-platform Java ME/Android apps.

E.5.1.1 Separating platform-dependent & -independent code

When developing an app that targets both platforms[44] with an intention to reuse (or rather share) as much code as possible, it is advisable to organise the source code in 3 codebases: two for the platform-specific parts of each app variant, and a third, shared one containing a *platform agnostic* implementation of the main behaviour of the app.

To decide whether or not a piece of code belongs in the shared codebase we propose the general rule stated in box E.1.

Platform agnosticism

Exception 1 in box E.1 follows directly from our observations on the differences between the Java ME CLDC/MIDP and Android platforms, as discussed in section E.4.

[44] Or refactoring and existing Java ME app to support Android as well.

All code should be shared *unless* one of the following exceptions applies:

Ex. 1: it cannot be expressed in a *platform agnostic* manner, because either:

- **Ex. 1a:** it is directly tied to platform-specific class libraries;
- **Ex. 1b:** it uses language constructs that are unsupported by one of the platforms;
- **Ex. 1c:** it depends on platform-specific system properties;

Ex. 2: it implements a platform/app-specific feature which cannot be feasibly or meaningfully generalised.

Box E.1: Rule for inclusion in the shared codebase

With *platform-specific class libraries*, as mentioned by exception 1a, we mean any class library which is not supported on both platforms. The only place where the platforms' class libraries overlap is at the fundamental, Java SE/Harmony-derived classes and interfaces, respectively specified by CLDC and provided by the Android Core Libraries (ACL)[45]. In terms of included classes/interfaces, as well as methods and fields thereof, CLDC's Java SE subset is smaller than and completely encompassed by ACL's Harmony subset, as illustrated by figures E.4 and E.5 on page 337. Hence compatibility with CLDC imposes the most limiting requirement. Therefore exception 1a can be rewritten as follows:

Ex. 1a*: it depends on[46] classes/interfaces which are outside the shared codebase itself *and* outside CLDC's Java SE subset; *or* it calls methods or accesses fields which are unavailable in CLDC's Java SE subset, despite being members of the original Java SE classes/interfaces included in that subset;

Box E.2: Refinement of exception 1a from box E.1

The situation created by exception 1a* is illustrated once more by figure E.7, in which the sets represent the entirety of supported class libraries of each platform. Any shared code dependency on a class, interface, method or field outside the sets' intersection would break either one of the apps. For example, relying on `java.util.ArrayList` – a Java SE-derived collection class included in ACL's Harmony subset but not in CLDC's Java SE subset – would break the Java ME app. Likewise a reference to `javax.microedition.midlet.MIDlet` – the superclass of all MIDlets – would break the Android app. In summary, the main test of exception 1a* is successful compilation of the

[45] We should note that this comparison is based on v1.1.1 of CLDC. This allows us to ignore the overlap between MIDP and ACL, because since v1.1.1 the CLDC specification includes the trio of Java SE-derived classes specified by MIDP up to v2.1 (as explained in footnote 26 on page 328).

[46] By "depend on" we mean any statically inferable relationship among classes, interfaces or their instances.

shared codebase against the class libraries of both platform SDKs. Because exception 1a* forces the shared codebase to only rely on fundamental Java classes, its contents can be seen as a “pure Java” implementation of the app’s core functionality.

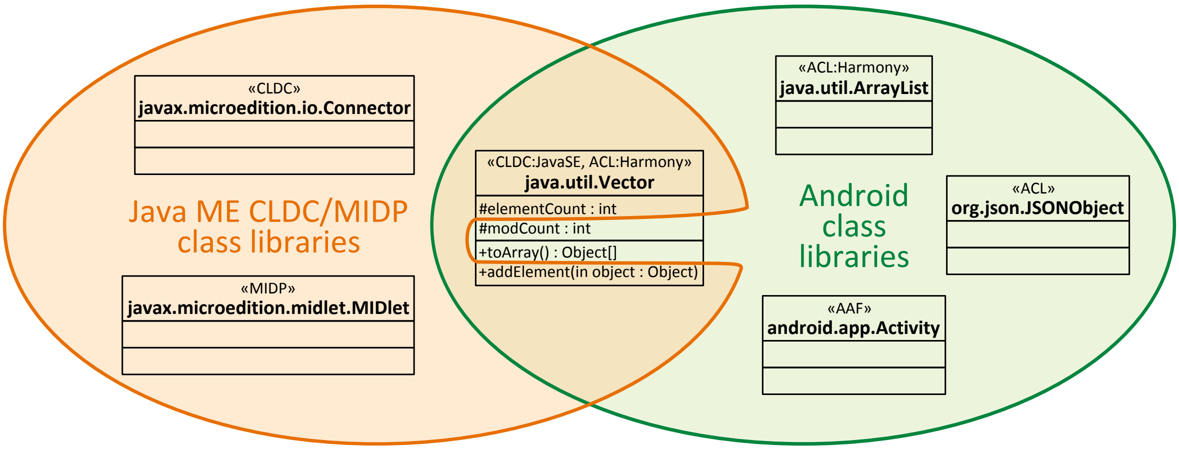


Figure E.7: The sets of Java ME CLDC/MIDP and Android class libraries and their intersection, showing some example classes

Exception 1b relates to the fact that, in order to be executable on CLDC devices, the code has to comply with the 2nd edition of the Java language specification [228] – whereas Android apps can be written in Java as defined in the 3rd edition of the specification [229]. Hence, third-edition features – such as generics, annotations or `foreach` loops – cannot be used in the Java ME-specific and in the shared codebase.

In Java, *system properties* are key-value associations that are queried with the `System.getProperty()` method. This mechanism allows programs to obtain information about the runtime environment, operating system, hardware capabilities, etc. This is especially necessary on Java ME. As stipulated by exception 1c, the shared code should not query system properties that are only supported by one of the platforms. While querying a non-existent system property does not cause compilation errors, and does not necessarily crash the program – unless the programmer forgets to check for a `null` value – the querying of platform-specific system properties is best done in platform-specific code, possibly upon request of a class in the shared codebase.

Fortunately only the shared codebase is restricted by these exceptions. It is obviously no problem to use any class/interface, or query any system property, supported by Java ME CLDC/MIDP in the Java ME-specific codebase, or to use any class/interface supported by Android, as well as third edition language features, in the Android-specific codebase.

Platform-specific features

Exception 2 in box E.1 applies to app features that were kept out of the shared codebase because they are closely tied to properties or capabilities of one of the platforms.

E.5.1.2 Relying on platform-specific APIs

Since the platforms' application frameworks – i.e. MIDP+Optional Packages vs. AAF – are incompatible, things like UI construction, network communication, audio/video playback or recording, use of positioning technology, sampling of various sensors, file system access, etc., all require class libraries/APIs outside of the intersection in figure E.7. In other words, such things cannot be expressed in "pure Java" alone.

However, as we have demonstrated in NoiseTube Mobile (see chapter 6), with a properly designed architecture it is possible to specify more or less all defining behaviour on an abstract level in the shared codebase, while leaving any concrete, platform-specific details to be filled in by the platform-specific codebases. Such a design can be based on the *Abstract Factory* design pattern, as described by the so-called *Gang of Four* (GoF) in their seminal *Design Patterns* book [198: pp. 87–95]. Figure E.8 illustrates the structure of this design pattern and mentions to which codebase each class belongs.

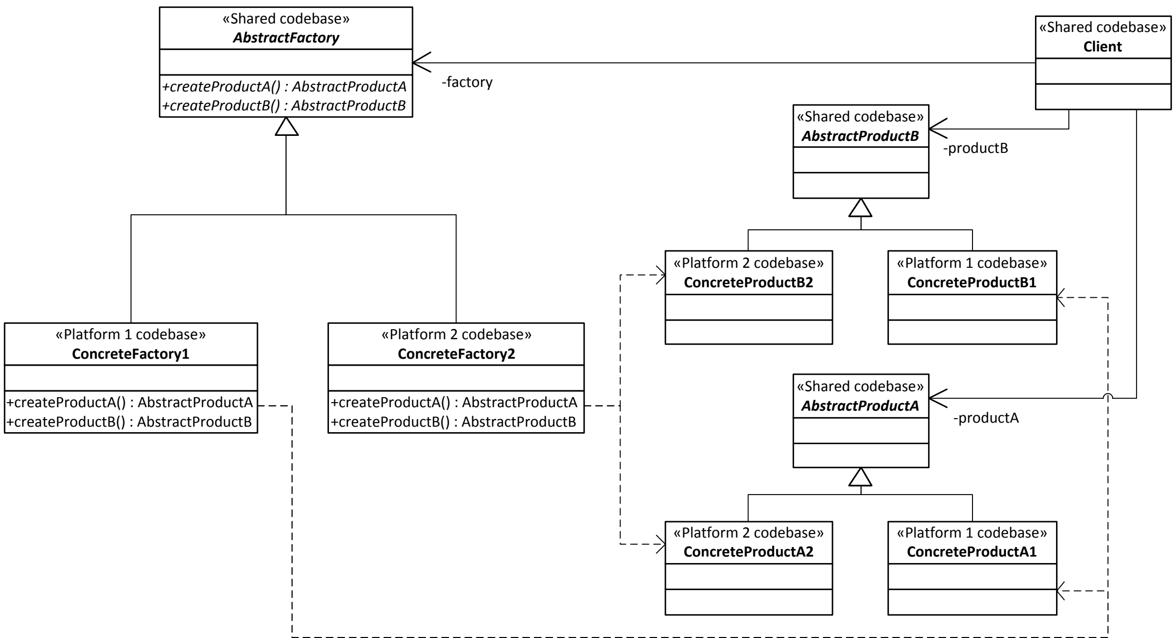


Figure E.8: UML class diagram illustrating the *Abstract Factory* design pattern

In order to make the *factory* instance(s) globally accessible without passing it/them around the *Singleton* pattern, also described by the GoF [198: pp. 127–134], may be used. Refer to section 6.5 for concrete examples of how these patterns can be applied.

E.5.2 Dealing with device variability

Every mobile phone model is different: they come in different shapes, have different feature sets, carry different hardware and software components and employ different user-input mechanisms. Furthermore model lifecycles are short (typically ≤ 1 year). Although mobile platforms and frameworks provide powerful abstractions to help developers deal with it, device variability remains – in our opinion – one of the most underestimated challenges of mobile app development. While many device details can be safely ignored, some differences, both in terms of soft- and hardware, have to be taken into account because they can undermine the robustness of apps. Even when targeting a single platform, developers of complex (e.g. relying on multiple APIs and/or device features), widely-deployed apps are likely to run into difficulties due to inconsistencies between devices of different vendors and even between products from single vendors. While many differences are documented in one way or another, most problems only come to light by testing the app on a range of different devices. Whether through study of vendor documentation or through testing, gathering information on device variations, determining if they (could) cause problems, and finding solutions if they do, is often a time-consuming and complicated task which strains the development and maintenance of mobile apps.

Our experience with the development of NoiseTube Mobile (see chapter 6) has shown that device variability is especially problematic on Java ME CLDC/MIDP.

E.5.2.1 Java ME CLDC/MIDP

Despite the fact that the Java ME CLDC/MIDP platform is/was supported across multiple (smart)phone brands and operating systems, the « *Write once, run anywhere* » mantra (see section E.3) should be taken with some caveats. Due to the sheer variety of MIDlet-supporting phones sold in the past decade – with many subtle and less subtle differences – it can be complicated to deploy a MIDlet across devices of varying brands and models. Of course this also depends on the concrete set of APIs and device features a MIDlet relies on. Generally speaking, the main things MIDlet developers should keep in mind are:

Platform component versions

The platform's components – CLDC, MIDP, and the optional packages – are governed by separate specifications (JSRs), most of which were updated a few times. Support for new specification versions or new optional packages, has typically appeared in new devices (after some delay), but old devices were seldom updated. Moreover, some devices may support additional vendor APIs, which are also updated from time to time. To tell in advance if a MIDlet can be expected to work on a particular device two bits of information are needed. On the one hand, the minimal versions of all platform components and possibly vendor APIs required by the

MIDlet must be known. On the other, vendor documentation must be consulted to know which specification(version)s are supported on the device in question. If there is a match[47], the MIDlet *could* work on that device.

Platform implementation differences

Compliance with the required specifications is no guarantee for proper MIDlet functioning because, in a sense, the platform only exists on paper. That is to say there is no single, common implementation of its specifications. Instead, several parties have independently developed runtime environments (REs). Despite adhering to the same, formal specifications there are many subtle differences between REs shipped on devices of different vendors and in some cases even between those found on different products of single vendors. The mean reason is that the specifications themselves leave some room for interpretation and occasionally declare specific capabilities as optional (e.g. support for audio recording is optional in MMAPI [283]). To avoid complications caused by this second source of differences, developers should carefully study those documentation [387, 487] sections which cover the specifics of a vendor's implementation of relevant specifications. Furthermore, MIDlets can be programmed – typically by querying system properties – to learn about a device's capabilities at runtime (e.g. whether audio recording is supported).

Undocumented limitations & bugs

Arguably the most annoying type of differences are undocumented limitations, peculiarities and outright bugs present in the REs found on some (or all) devices of particular vendors. Some examples are discussed in section 6.5.5.

Hardware features

Occasionally it is a good idea to let a MIDlet adapt itself to hardware features, such as the resolution of the screen or whether or not it is a touchscreen.

E.5.2.2 Android

The Android platform is somewhat less affected by device variability. This is in part due to the fact that the platform supports a vast amount of features "out of the box" (partially eliminating the need for vendor APIs or additional libraries). However the rapid succession of Android platform versions – which manufacturers have a hard time following – and the growing variety of device form factors may be (or become) problematic.

[47] On devices released since 2008, CLDC and MIDP versions rarely pose problems but optional package support can still be problematic (see section E.3.3).

E.5.3 Dealing with network failures

Data connections over cellular networks are often volatile. An obvious cause for failures is when the device moves out of the range. This can happen when traveling through an area with sparse network coverage (or no coverage at all). But even in well-covered (urban) areas there are places where the network can be temporarily or permanently unavailable (e.g. buildings with thick walls, underground areas like tunnels or subway stations, and other so-called *dead zones*). Even when contact with the network is not lost – i.e. phone calls can be made and SMS messages can be sent/received – data connections may be interrupted or slowed down due to network congestion, varying signal strength and other factors. Short interruptions can also happen during handoffs between different network types[48]. While most interruptions or delays are short in duration they happen more than people are (or want to be) aware of. Hence, to avoid a frustrating user experience, apps that access the Internet should be resilient to network failure, and ideally not bother the user with it, unless perhaps when the problem turns out to be permanent. As we demonstrated in chapter 6 one way to increase resilience to network failure is to introduce a caching mechanism.

[48] For example, today's smartphones often autonomously switch between cellular data connections and faster, and usually cheaper, Wi-Fi networks.

Appendix F

Questionnaire for campaign participants & other NoiseTube users

In order to evaluate the NoiseTube system from the perspective of end-users we assembled a questionnaire. So far this questionnaire has only been used with the member of the Ademloos activism group who participated in the noise mapping experiments discussed in section 7.3. The main findings of that particular user study are discussed in section 7.3.7. However the questionnaire itself is intended as a first step towards a more general evaluation tool that (when fine-tuned and translated into English or other languages) could be used with larger audiences of NoiseTube users.
The questionnaire in it its current form – in Dutch and with some questions that are only relevant to the volunteers of Ademloos – is included on the following pages.

Vrije Universiteit Brussel

NoiseTube

Beste deelnemer van de NoiseTube meetcampagne(s),

We willen jullie nogmaals bedanken voor jullie deelname aan de onderzoeksproject.

Om jullie ervaringen met de NoiseTube applicatie beter te begrijpen en de nodige aanpassingen te kunnen doen, vragen we jullie nog een half uur tijd om deze vragenlijst in te vullen.

Het is belangrijk dat je deze vragenlijst individueel invult. Er zijn geen juiste of foute antwoorden, het is uw persoonlijke mening die telt.

We beginnen met enkele persoonlijke gegevens

1. Postcode : ....
2. Hoe lang woont u daar al? Sinds ....
3. Geslacht: O Man O Vrouw
4. Geboortejaar: ....
5. Hoogste behaalde diploma:
   - O diploma lager onderwijs
   - O diploma middelbaar onderwijs
   - O diploma hoger of universitair onderwijs
   - O post–universitaire diploma
6. Studeert u nog?

   O ja O nee
7. Huidige of laatste beroep: ....

Meetcampagnes

8. Aan welke meetcampagne(s) nam u deel. *Duid de passende campagne(s) aan.*

O 5-9 juli 2010; 21-22u	(Fase 1 - week 1)
O 12-16 juli 2010; 8u30-9u30	(Fase 1 - week 2)
O 15-20 november 2010; vrij gekozen momenten, 1u per dag	(Fase 2)

Uw verwachtingen over geluidskaarten

De geluidsmetingen waaraan u hebt bijgedragen zullen worden voorgesteld op kaarten. We willen graag weten wat uw verwachtingen zijn over deze kaarten. Vandaar de volgende vragen:

9. Welke kenmerkten verwacht u ideaal gezien aan te treffen op een kaart (of kaarten) over geluidsmetingen?
Kies per kenmerk de best passende categorie op de schaal van heel belangrijk tot heel onbelangrijk.

Kenmerken op geluidskaart(en):	**Heel belangrijk**	**Belangrijk**	**Belangrijk noch onbelangrijk**	**onbelangrijk**	**Heel onbelangrijk**
Straat met naam					
Gebouwen					
Functie van plaatsten of gebouwen (vb: school)					
Gemeten geluidsniveau					
Variatie geluidsniveau naar plaats					
Variatie geluidsniveau naar tijd					
Betrouwbaarheid van de geluidsniveau metingen					
Oorzaken van de gemeten geluiden					

10. Zijn er nog andere kenmerken die u nuttig zou vinden?

………

………

11. Welke weergave op de kaart heeft je voorkeur voor het **gemeten geluidsniveau**? *Kies één van beide mogelijkheden.*

O Weergave in klassen (vb: van 55 tot 60 dB, van 60 tot 65, enz.)

O Exacte weergave van elke meting (vb: 55,4 dB; 67,2 dB)

12. Welke weergave op de kaart heeft je voorkeur qua **variatie in geluidsniveau**? *Kies één van beide mogelijkheden.*

O piekwaarde/hoogste geluidssterkte

O gemiddelde geluidssterkte

13. Welke weergave op de kaart heeft je voorkeur qua **variatie in tijd** ? *Kies één van beide mogelijkheden.*

O geluidssterkte per uur

O geluidssterkte per dagdeel (vb: ochtend, middag, avond, nacht)

14. Welke weergave op de kaart heeft je voorkeur qua **oorzaken van geluid** ? *Kies één van beide mogelijkheden.*

O Oorzaken van geluid door functies van gebied te geven (vb: "fabriek", …)

O Oorzaken van geluid vastgesteld ter plaatse (vb: "wegenwerken", …)

15. Welke weergave op de kaart heeft je voorkeur qua **betrouwbaarheid geluidssterkte** ? *Kies één van beide mogelijkheden.*

O Betrouwbaarheid geluidssterkte door weer te geven **wie** meting deed

O Betrouwbaarheid geluidssterkte door **aantal metingen** weer te geven

2 / 8

Motivaties en ervaringen bij het meten

We willen graag weten wat u motiveerde om deel te nemen aan de meetcampagne(s) en hoe u het meten en het daarvoor gebruikte toestel ervaren hebt.

Laten we eerst in gedachten teruggaan naar het moment waarop u toezegde aan de meetcampagne deel te nemen.

16. Waarom nam u deel aan het project? Wat was u motivatie?
Kies per mogelijke motivatie de best passende categorie op de schaal van heel belangrijk tot heel onbelangrijk.

Motivatie:	**Heel belangrijk**	**Belangrijk**	**Belangrijk noch onbelangrijk**	**Onbelangrijk**	**Heel onbelangrijk**
Persoonlijk ervaren hinderlijke geluidsoverlast					
Algemene bezorgdheid geluidsoverlast					
Steun actiegroep					
Nieuwsgierigheid					
Interesse technologie					
Nuttige tijdsbesteding					
Leuke tijdsbesteding					
Steun wetenschappelijk onderzoek					

17. Had u andere redenen om deel te nemen aan het project? Zo ja, welke?

....

....

....

....

Laten we nu in gedachten teruggaan naar de week (of weken) waarin u deelnam aan een meetcampagne.

18. Wat vond u fijn aan het meten? *Geef de 3 belangrijkste redenen.*

a. ....

b. ....

c. ....

19. Wat vond u lastig aan het meten? *Geef de 3 belangrijkste redenen.*

a. ....

b. ....

c. ....

3 / 8

20. Geef aan wat u van de volgende aspecten bij het uitvoeren van meetsessies met de NoiseTube applicatie vond:
Kruis per aspect best passende categorie (zeer goed tot zeer slecht) aan.

Aspecten :	**Zeer goed**	**Goed**	**Goed noch slecht**	**Slecht**	**Zeer slecht**
Duidelijkheid van de instructies die gegeven werden om de meetsessie correct uit te voeren					
De duidelijkheid van de algemene begeleiding door de onderzoekers					
De duidelijkheid van de begeleidende informatie					
Keuze van het te wandelen traject/gebied van de meetsessies					
Het gekozen tijdstip van de meetsessies (fase 1)					
Het starten van een nieuwe meetsessie op het telefoontoestel					
De informatie op het scherm va het toestel tijdens een meetsessie					
De informatie op het scherm va het toestel na het beëindigen van een meetsessie					
Het telefoontoestel zelf					

21. Had je graag kunnen aanduiden wat de **oorzaak van een gemeten geluid** was (vb: “hier was file”, “ambulance”)?

O ja O misschien O nee

22. Had je graag via een website commentaar kunnen geven over de metingen gedaan **door uw groep?**

O ja O misschien O nee

23. Had je graag via een website comment kunnen geven op de metingen gedaan **door andere gebruikers van NoiseTube**?

O ja O misschien O nee

24. Was u op een bepaald moment (of momenten) bezorgd om uw **privacy** tijdens het gebruiken van NoiseTube?

O ja O nee

Zo ja, waarom?

………

………

25. Hebt u nog **suggesties en opmerkingen** ter verbetering van de NoiseTube applicatie zoals u ze testte?

………

………

………

………

………

4 / 8

Wat kan en wil men doen met de door jullie verzamelde geluidsmetingen?

In de laatste sectie van deze vragenlijst, willen we een aantal vragen stellen over uw visie op het verzamelen van dit type van data door burgers.

We gebruiken in deze sectie volgende termen:

Met **burgers** bedoelen we u en ik, als inwoner van een stad, een regio en een land.

We gebruiken ook het woord **professionele expert**, daarmee bedoelen we iemand die een erkende opleiding heeft genoten om metingen uit te voeren en te interpreteren, en die kan aangesteld worden door iedereen die zijn of haar diensten inhuurt.

Ten slotte verwijzen we naar **officiële data of metingen**, om de metingen georganiseerd door een overheid aan te duiden, zoals de metingen vandaag de dag gebeuren.

26. Welke **voordelen** ziet u aan het door burgers laten verzamelen van geluids- of andere milieugegevens?

a. ……

b. ……

c. ……

27. Welke **nadelen** ziet u aan het door burgers laten verzamelen van geluids- of andere milieugegevens?

a. ……

b. ……

c. ……

28. Door wie mogen deze gegevens geraadpleegd worden? *Kruis aan. Meerdere antwoorden zijn mogelijk.*

O Iedereen die een deel van de data verzameld heeft
O Alle geïnteresseerde burgers
O Professionele experts
O Officiële instanties die overheidsbeleid ondersteunen
O Andere: Welke? ………

29. Waarom zou u **zelf** deze gegevens willen gebruiken?

……

……

……

……

……

30. **Hoe** zou u dat doen?

……

……

……

……

……

31. Zou je opnieuw meedoen aan een **wetenschappelijke studie** over geluidsmetingen?

a. O ja O misschien O nee

b. Waarom wel of niet? ………

32. Zou u op **eigen initiatief** willen verder werken meten met dit soort meettoestel?

a. O ja O misschien O nee

b. O alleen O in groep O beide

c. Waarom wel of niet? ………

33. Geef voor volgende stellingen rond geluidsmetingen aan welke categorie uw mening het beste weergeeft. *Per stelling is er slechts één antwoord mogelijk.*

Stellingen ivm geluidsmetingen:	**Helemaal eens**	**Eens**	**Eens noch oneens**	**Oneens**	**Helemaal oneens**
1. Er zijn voldoende officiële geluidsmetingen.					
2. Officiële metingen zijn niet representatief voor wat mensen dagelijks ervaren.					
3. Enkel officiële procedures zorgen voor representatieve metingen.					
4. Enkel professionele meetapparatuur zorgt voor representatieve metingen.					
5. Enkel professioneel opgeleide experts zorgen voor representatieve metingen.					
6. Enkel de professionele experts geven een waardevolle interpretatie aan geluidsmetingen.					
7. Door burgers verzamelde data geeft de mogelijkheid om officiële visie aan te vechten.					
8. Door burgers verzamelde data geeft meer kennis van eigen problematiek.					
9. Door burgers verzamelde data geeft de mogelijkheid om oorzaken te begrijpen.					
10. Door burgers verzamelde data geeft de mogelijkheid om door de overheid onderbelichte tijdstippen, plaatsen te bestuderen.					
11. Door burgers verzamelde data helpt niet want onwetenschappelijk en gevaarlijk.					

Stellingen ivm geluidsmetingen (vervolg):	**Helemaal eens**	**Eens**	**Eens noch oneens**	**Oneens**	**Helemaal oneens**
12. Met door burgers verzamelde data kan je makkelijker media aandacht krijgen voor de problematiek.					
13. Door burgers verzamelde helpt andere burgers aan te zetten om samen actie te voeren.					
14. Meer meetgegevens gaan de oorzaak niet wegnemen, de overheid kent het probleem al maar onderneemt te weinig.					
15. Mensen gaan enkel de problemen in kaart brengen, waardoor de gegevens nooit een betrouwbaar beeld zullen weergeven van de gehele situatie.					
16. De kwaliteit van door burgers verzamelde data is niet controleerbaar.					
17. Door burgers verzamelde data is minder accuraat want met ze beschikken niet over professioneel materiaal.					
18. Door burgers verzamelde data is minder accuraat want ze missen een professionele opleiding.					
19. Zelfs minder accurate data kan, in grote hoeveelheden, waardevol zijn om relatieve verschillen of patronen te ontdekken.					
20. Door burgers verzamelde data en officiële metingen vullen elkaar best aan om tot goede kaarten te komen.					

7 / 8

Tot slot
We zijn aan het einde van deze vragenlijst. Nogmaals hartelijk bedankt! Indien u nog suggesties of opmerkingen heeft, kan u die hier altijd nog kwijt hieronder.

Appendix G

Dissemination & impact

Here we give an overview of the dissemination of our work and the impact it has and continues to have. This includes scholarly and other publications, contributions at events, and media mentions.

G.1 Publications

Portions of the work discussed in this dissertation have been previously covered in the following publications:

Journal papers

- Nicolas Maisonneuve, Matthias Stevens, and Bartek Ochab. "Participatory noise pollution monitoring using mobile phones". In: *Information Polity* 15.1-2 (Aug. 2010): *Special Issue on "Government 2.0: Making Connections between Citizens, Data and Government"*, pp. 51–71.
  ISSN: 1570-1255
  DOI: 10.3233/IP-2010-0200
  URL: http://soft.vub.ac.be/Publications/2010/vub-tr-soft-10-14.pdf

- Ellie D'Hondt, Matthias Stevens, and An Jacobs. "Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring". Preprint. Submitted to *Pervasive and Mobile Computing* (ISSN: 1574-1192). Conditionally accepted. Dec. 2011.
  URL: http://www.noisetube.net/publications/partnoisemaps.pdf

Conference papers

- Nicolas Maisonneuve, Matthias Stevens, Maria E. Niessen, Peter Hanappe, and Luc Steels. "Citizen Noise Pollution Monitoring". In: *dg.o '09 – Proceedings of the 10th Annual International Conference on Digital Government Research (Puebla, Mexico; May 17-21, 2009). Social Networks: Making Connections between Citizens, Data and Government*. Ed. by Soon Ae Chun, Rodrigo Sandoval, and Priscilla Regan. ACM International Conference Proceeding Series, vol. 390. Digital Government Society of North America, pp. 96–103.
  ISBN: 978-1-60558-535-2
  URL: http://portal.acm.org/citation.cfm?id=1556176.1556198

- Nicolas Maisonneuve, Matthias Stevens, Maria E. Niessen, and Luc Steels. "NoiseTube: Measuring and mapping noise pollution with mobile phones". In: *Information Technologies in Environmental Engineering. Proceedings of the 4th International ICSC Symposium (ITEE 2009; Thessaloniki, Greece, May 28-29, 2009)*. Ed. by Ioannis N. Athanasiadis, Pericles A. Mitkas, Andrea E. Rizzoli, and Jorge Marx Gómez. Environmental Science and Engineering. Springer Berlin Heidelberg, pp. 215–228.
  ISBN: 978-3-540-88350-0
  ISSN: 1863-5520
  DOI: 10.1007/978-3-540-88351-7_16
  URL: http://soft.vub.ac.be/Publications/2009/vub-prog-tr-09-08.pdf

- George Drosatos, Pavlos S. Efraimidis, Ioannis N. Athanasiadis, Ellie D'Hondt, and Matthias Stevens. "A privacy-preserving cloud computing system for creating participatory noise maps". In: *COMPSAC 2012 – Trustworthy Software Systems for the Digital Society. IEEE Signature Conference on Computers, Software & Applications (Izmir, Turkey, July 16-20, 2012)*. Accepted for publication.

Workshop papers

- Nicolas Maisonneuve, Matthias Stevens, and Luc Steels. "Measure and map noise pollution with your mobile phone". Instructable. In: *DIY::HCI. A Showcase of Methods, Communities and Values for Reuse and Customization*. Ed. by Leah Buechley, Eric Paulos, Daniela Rosner, and Amanda Williams. Proceedings of the "DIY for CHI" workshop held at CHI '09, the 27th International Conference on Human Factors in Computing Systems (April 4-9, 2009, Boston, MA, USA), pp. 78–82.
  URL: http://soft.vub.ac.be/Publications/2009/vub-prog-tr-09-03.pdf

- Jorge Vallejos, Matthias Stevens, Ellie D'Hondt, Nicolas Maisonneuve, Wolfgang De Meuter, Theo D'Hondt, and Luc Steels. "Context-aware Resource Sharing for People-centric Sensing". In: *First International Workshop on Software Research and Climate Change (WSRCC-1), part of Onward! 2009 (co-located with OOPSLA 2009) in Orlando, FL, USA, October 26, 2009*. Ed. by Steve Easterbrook.
  URL: http://www.cs.toronto.edu/wsrcc/WSRCC1/Papers.html

- Matthias Stevens and Ellie D'Hondt. "Crowdsourcing of Pollution Data using Smartphones". In: *Proceedings of the Workshop on Ubiquitous Crowdsourcing, held at Ubicomp '10 (Copenhagen, Denmark, September 26-29, 2010)*.
  URL: http://soft.vub.ac.be/Publications/2010/vub-tr-soft-10-15.pdf

Demos and Posters

With published abstract:

- Ellie D'Hondt and Matthias Stevens. "Community memories for sustainable urban living". Poster abstract. In: *INNOV 2010 – Proceedings of the 1st International Multi-Conference on Innovative Developments in ICT (Athens, Greece, July 29-31, 2010)*. Ed. by Boris Shishkov, George A. Tsihrintzis, and Maria Virvou, pp. 77–80. Poster displayed at ICGREEN 2010 – International Conference on Green Computing (part of INNOV 2010).
  ISBN: 978-989-8425-15-7
  URL: http://soft.vub.ac.be/Publications/2010/vub-tr-soft-10-16.pdf

- Ellie D'Hondt and Matthias Stevens. "Participatory noise mapping". Demo abstract. In: *Demo Proceedings of the 9th International Conference on Pervasive Computing (Pervasive 2011), San Francisco, CA, USA, June 12-15, 2011*. Ed. by Rafael 'Tico' Ballagas and Daniela K. Rosner, pp. 33–36.
  URL: http://soft.vub.ac.be/Publications/2011/vub-soft-tr-11-03.pdf

Without abstract:

- Nicolas Maisonneuve, Matthias Stevens, and Bartek Ochab. "NoiseTube: Citizen Noise Pollution Monitoring". Poster displayed at the Sony CSL Paris Open House 2009 (Paris, France, 8-9 October, 2009).
  URL: http://soft.vub.ac.be/Publications/2009/vub-prog-tr-09-11.pdf

- Matthias Stevens and Ellie D'Hondt. "Participatory Sensing for Sustainable Urban Life". Poster displayed at the MoVES project annual meeting (Université catholique de Louvain, Louvain-la-Neuve, Belgium, December 17, 2009).
  URL: http://soft.vub.ac.be/Publications/2009/vub-soft-tr-09-05.pdf

- Matthias Stevens and Ellie D'Hondt. "Participatory Sensing for Sustainable Urban Life". Poster displayed at the Dag van de Doctorandi event (VUB, Brussels, Belgium, May 28, 2010).
  URL: http://soft.vub.ac.be/Publications/2009/vub-soft-tr-09-05.pdf

Theses

- Sander Bartholomees. "Context-Aware Urban Sensing". Promoted by Wolfgang De Meuter. Advised by Matthias Stevens and Ellie D'Hondt. MA thesis. Vrije Universiteit Brussel, Faculteit Wetenschappen en Bio-ingenieurswetenschappen, Vakgroep Computerwetenschappen, June 2011. 128 pp.
  URL: http://brussense.be/Pubs/files/Bartholomees2011.pdf

Popular science

- Matthias Stevens and Ellie D'Hondt. "NoiseTube & BrusSense: Participatory sensing for sustainable urban life". In: *We Can Change the Weather: 100 Cases in Energy Efficacy to Start Doing So*. Ed. by Marleen Wynants and Sara Engelen. Crosstalks. ASP-VUBPress, 2010, pp. 106–107.
  ISBN: 978-9054876922
  URL: http://crosstalks.vub.ac.be/publications/changetheweather/intro.html

As of 2012-04-13 and according to *Publish or Perish* [248], the scholarly articles listed above have been cited 60 times (for a total of 4 cited papers, good for an *h*-index of 4).

G.2 Event contributions

From 2008 to 2012 we have presented, demonstrated or discussed portions of the work covered in this dissertation at a variety of events, aimed at academic (A), industrial (I), governmental (G), artistic (R), or general public (P) audiences.

2008

R/P **Cartographies parallèles, rencontre avec Christian Nold** (organisation: Villes 2.0)
La Cantine, Paris, June 19, 2008.
Contribution: Matthias Stevens & Nicolas Maisonneuve took part in discussion.

2009

R **GeoTales – Locative media workshop**
iMAL, Brussels, February 23, 2009.
Contribution title: *Realisation of Locative Experiences using Smartphones* (by Matthias Stevens).

A **DIY for CHI: Methods, Communities, and Values of Reuse and Customization**, workshop held at **CHI 2009, 27th Annual ACM SIGCHI Conference**
Boston, MA, USA, April 4-9, 2009.
Contribution title: *The NoiseTube project: Measure and map noise pollution with your mobile phone* (presented by Matthias Stevens).

A **ITEE 2009, 4th International ICSC Symposium on Information Technologies in Environmental Engineering**
Thessaloniki, Greece, May 28-29, 2009.
Contribution title: *NoiseTube: Measuring and mapping noise pollution with mobile phones* (presented by Nicolas Maisonneuve).

A/I/P **Sony CSL Paris Open House**
Paris, France, October 8-9, 2009.
Contributions: presentation *Participatory Mapping and Social Networking for a Sustainable World* (by Luc Steels); poster and demo session (by Nicolas Maisonneuve & Matthias Stevens).

A/I **Energy Efficiency: Facing the Facts & Learning to Cooperate; Workshop #6: Limits to Growth** (organisation: Crosstalks, VUB)
Brussels, November 27, 2009.
Contribution title: *NoiseTube: Participatory Sensing for Sustainable Urban Life* (by Matthias Stevens).

A/I **"Are you ready for the Internet of Things?"** (organisation: Council)
iMAL, Brussels, December 4, 2009.
Contribution title: *NoiseTube: Participatory Sensing for Sustainable Urban Life* (by Matthias Stevens).

2010

G/P **Stadspiratie: Congres stedelijk netwerken** (organisation: VGC)
Brussels, 25 February 2010.
Contribution title: *NoiseTube: Participatory Sensing for Sustainable Urban Life* (presentation and demo, by Ellie D'Hondt & Matthias Stevens).

A **Campustalks Session #06** (organisation: Crosstalks, VUB)
Université Libre de Bruxelles, Brussels, 1 April 2010.
Contribution title: *Participatory Sensing for Sustainable Urban Life* (presented by Matthias Stevens).

A **Workshop on Understanding, Modelling and Measuring Soundscapes**
Universiteit Gent, Gent, 27-28 April 2010.
Contribution title: *NoiseTube: Participatory Noise Pollution Monitoring using Mobile Phones* (presented by Ellie D'Hondt).

G **Studiedag gezondheidsbeleid** (organisation: Scholengroep Brussel)
Brussels, May 6, 2010.
Contribution: NoiseTube demonstration session (by Ellie D'Hondt).

I/A **Flanders Smart Hub goes Pecha Kucha**
Museum M, Leuven, May 27, 2010.
Contribution title: *Participatory Sensing for Sustainable Urban Life* (invited talk by Matthias Stevens).

A **Dag van de Doctorandi**
VUB, Brussels, May 28, 2010.
Contribution title: *Participatory Sensing for Sustainable Urban Life* (poster).

R/P **E-Culture Fair 2010** (organisation: BAM)
Dortmund, Germany, August 23-25, 2010.
Contribution title: *NoiseTube: Participatory Sensing for Sustainable Urban Life* (demo and talk by Matthias Stevens).

A **ICGreen 2010, 1st International Conference on Green Computing**
Athens, Greece, August 29-31, 2010.
Contribution title: *Community memories for sustainable urban living* (poster).

A/P/G/I/R **Book Release "We can change the weather: 100 cases of changeability"** (organisation: Crosstalks, VUB).
Brussels, September 21, 2010.
Contribution: Matthias Stevens & Ellie D'Hondt talked about their contribution to the book.

A **UbiCrowd 2010, 1st International Workshop on Ubiquitous Crowdsourcing**, held at **UbiComp 2010, 12th ACM International Conference on Ubiquitous Computing**
Copenhagen, Denmark, 26-29 September 2010.
Contribution title: *Crowdsourcing of Pollution Data using Smartphones* (presented by Matthias Stevens).

A/I **2nd yearly Software Languages Lab Event**
VUB, Brussels, December 3, 2010.
Contribution title: *The human touch: Environmental sensing and human-computer interaction* (presented by Lode Hoste & Matthias Stevens).

2011

A/G/R/P **La Semaine du Son, 1st Brussels edition** (organisation: Halolalune Production)
Brussels, January 24-30, 2011.
Contributions: presentation *Ecouter et mesurer les bruits dans la ville* and demo *Parcours de sensibilisation et de mesures sonores dans la ville* (with Matthias Stevens & Ellie D'Hondt), January 29, 2011.

A **Invited seminar** (organisation: Dept. of Civil, Environmental and Geomatic Engineering, University College London)
London, UK, March 1, 2011.
Contribution title: *NoiseTube: A multi-scale participatory approach to noise monitoring and mapping* (by Matthias Stevens).

A **ESF Workshop on The Internet of Things for a Sustainable Future**
Vielsalm, Belgium, 9–13 May 2011.
Contributions: organised by Ellie D'Hondt, co-organised by Matthias Stevens, presentation *NoiseTube & beyond: a participatory approach for pollution mapping* (by Matthias Stevens).

A/I **Pervasive 2011, 9th International Conference on Pervasive Computing**
San Francisco, CA, USA, June 12-15, 2011.
Contribution title: *Participatory noise mapping* (demo by Ellie D'Hondt).

A **BEST Brussels Summer Course 2011: From Wireless Sensor Networks to the Internet of Things** (organisation: Board of European Students of Technology)
Brussels, July 20-30, 2011.
Contribution title: *Participatory noise mapping* (presented by Ellie D'Hondt, on July 28, 2011).

P/G **Lezing over geluidshinder & -meting** (organisation: Gemeente Zwijndrecht)
Zwijndrecht, Belgium, August 23, 2011.
Contribution: *Stop dat lawaai! Meet het eerst! Met je gsm bijvoorbeeld!* (with Ellie D'Hondt)

R/P **CityBeat project.** (co-produced by the Finnish Bioart Society and the European Public Art Centre)
Helsinki, Finland, August 26 – September 30, 2011.
Contribution: the NoiseTube technology was used as part of an art installation.

I/G/A **Round Table: Green ICT and ICT for Green** (organisation: Belgacom)
Brussels, September 2, 2011.
Contribution: Ellie D'Hondt took part in a panel discussion with 8 participants from industry and government.

A/I/P **Brussel innoveert!** (organisation: Brussels Hoofdstedelijk Gewest)
Exposition for innovative companies and institutions in the Brussels Region.
Woluwe Shopping Center, Brussels, October 16–30, 2011.
Contribution: With exposition stand and interactive NoiseTube demo.

A **Invited seminar** (organisation: Faculty of Urban Planning, Università Iuav di Venezia)
November 10, 2011, Venice, Italy.
Contribution title: *NoiseTube: participatory sensing for sustainable urban living* (by Ellie D'Hondt).

A/I **Research met en voor de geomatica-industrie v.2.0**
KaHo Sint-Lieven, Gent, December 13, 2011.
Contribution title: *Collaborative monitoring and mapping urban noise pollution* (invited talk by Matthias Stevens).

A/I/G **Campustalks #10: Brussel Leefbare Stad** (organisation Crosstalks, VUB)
KultuurKaffee, VUB, Brussels, December 20, 2011.
Contribution: *BrusSense: Participatory Sensing for Sustainable Urban Life* (by Ellie D'Hondt).

2012

A/G/R/P **La Semaine du Son, 2nd Brussels edition** (organisation: Halolalune Production)
Brussels, January 23-29, 2012.
Contributions: *Exposition NoiseTube*, January 24-26, vernissage on January 23; *Parcours de mesures sonores*, January 23 & 26; *Seminar: Noise mapping*, January 28 (presentation by Ellie D'Hondt & Marie Poupe, IBGE/BIM); and *Contest: Take part in noise mapping campaign and win a smartphone!*, January 1-29.

A/I/G **Kick-off meeting i-Scope project** (interoperable Smart City services through an Open Platform for urban Ecosystems)
January 23 & 25, 2012, VUB, Brussels.
Contribution: local organisation & presentation of BrusSense Team (by Ellie D'Hondt).

A/I/G/P **2nd London Citizen Cyberscience Summit** (organisation: Citizen Cyberscience Centre & UCL Excites)
London, February 16-18, 2012
Contributions: presentation *Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring* (by Ellie D'Hondt), February 17; workshop *Noise and the City*, February 17; Hackday challenge *What can YOU do with Noise Data?*, February 18.

G.3 Media mentions

The research discussed in this dissertation has been mentioned, either directly or indirectly, in the following online, printed or audio/visual media:

2009

Web Erica Westly. *Citizen Science: How Smartphones Can Aid Scientific Research*. Popular Mechanics. Mar. 13, 2009.
URL: http://www.popularmechanics.com/science/research/4308375.html
ARCHIVED: http://noisetube.net/publications/PopularMechanics_13-03-2009.pdf

Print Anil Ananthaswamy. "A crowd-sourced solution to noise pollution monitoring. Why anti-noise campaigners may learn to love cellphones". Feature. In: *The New Scientist* 204.2734 (Nov. 14, 2009), p. 26.
ISSN: 0262-4079
DOI: 10.1016/S0262-4079(09)62989-4
ARCHIVED: http://noisetube.net/publications/NewScientist2009-11-14.pdf

Web Anil Ananthaswamy. *Cellphone app to make maps of noise pollution*. NewScientist. Nov. 18, 2009.
URL: http://www.newscientist.com/article/mg20427346.900 (paywalled)
ARCHIVED: http://noisetube.net/publications/NewScientist.com_2009-11-18.pdf

Web 20 Minutes Online. *Les téléphones portables combattent la pollution sonore*. Nov. 20, 2009. In French.
URL: http://www.20min.ch/ro/multimedia/stories/story/23317149
ARCHIVED: http://noisetube.net/publications/20MinutesOnline2009-11-20.pdf

Radio Pure FM (RTBF). *On n'est pas des anges*. Nov. 23, 2009. Interview with Matthias Stevens. In French.
URL: http://www.rtbf.be/purefm/emission_on-n-est-pas-des-anges?id=258
ARCHIVED: http://noisetube.net/publications/InterviewRTBF.mp3

Print Elise Dubuisson. "Quand le téléphone traque le bruit. La VUB participe au projet international « NoiseTube »". Pollution. In: *Le Soir* 276 (Nov. 27, 2009), p. 31. In French.
ARCHIVED: http://noisetube.net/publications/LeSoir_27-11-2009.pdf

Web Elise Dubuisson. *Quand le GSM traque le bruit. La VUB participe au projet international « NoiseTube »*. Pollution. Le Soir. Nov. 27, 2009. In French.
URL: http://archives.lesoir.be/?action=nav&gps=740409
Archived: http://noisetube.net/publications/LeSoir.be_27-11-2009.pdf

Web La Dernière Heure. *Quand le téléphone traque le bruit*. Nov. 27, 2009. In French.
URL: http://www.dhnet.be/infos/societe/article/290629/quand-le-telephone-traque-le-bruit.html
Archived: http://noisetube.net/publications/LDH_27-11-2009.pdf

Web L'Echo. *Quand le téléphone traque le bruit (Le Soir)*. Nov. 27, 2009. In French.
URL: http://lecho.be/r/?t=1&id=8265286 (paywalled)
Archived: http://noisetube.net/publications/L'Echo_27-11-2009.pdf

Web La Libre Belgique. *Quand le téléphone traque le bruit (Le Soir)*. Nov. 27, 2009. In French.
URL: http://www.lalibre.be/toutelinfo/belga/124397/quand-le-telephone-traque-le-bruit-le-soir.html (offline)
Archived: http://noisetube.net/publications/LLB_27-11-2009.pdf

Web Le Vif/L'Express. *Quand le téléphone traque le bruit (Le Soir)*. Nov. 27, 2009. In French.
URL: http://levif.rnews.be/belga/generale/78-6-132656/quand-le-telephone-traque-le-bruit--le-soir-.html (offline)
Archived: http://noisetube.net/publications/LeVifL'Express_27-11-2009_(1).pdf

Web Le Vif/L'Express. *NoiseTube, ou comment mesurer le bruit ambiant avec son GSM*. Nov. 27, 2009. In French.
URL: http://focus.levif.be/loisirs/article-1194669681553.htm
Archived: http://noisetube.net/publications/LeVifL'Express_27-11-2009_(2).pdf

Web RTBF. *Quand le téléphone traque le bruit: Noise Tube*. Nov. 27, 2009. In French.
URL: http://www.rtbf.be/info/belgique/economie/quand-le-telephone-traque-le-bruit-le-soir (offline)
Archived: http://noisetube.net/publications/RTBF-INFO_27-11-2009.pdf

Print "VUB wil geluidsoverlast meten met gsm's". In: *Het Laatste Nieuws* (Nov. 28, 2009), p. 23. In Dutch.
Archived: http://www.vub.ac.be/infoover/media/uab20091128.html#doc5

2010

Print Peter Van Rompaey. "Mobiele telefoon als meettoestel". Junior onderzoeker. In: *Akademos* 13.1 (Feb. 2010), p. 13. In Dutch.
URL: http://www.vub.ac.be/downloads/akademosfebruari2010.pdf

TV France 2. *Télématin*. Mar. 11, 2010. Interview with Nicolas Maisonneuve. In French.
URL: http://telematin.france2.fr/?page=chronique&id_article=14987 (offline)
ARCHIVED: http://noisetube.net/publications/Telematin-20100311.mp4

Web De Morgen. *"Ruim 3.000 Antwerpenaren zullen voortijdig sterven door verkeerslawaai"*. Dossier Oosterweel. Mar. 22, 2010. In Dutch.
URL: http://demorgen.be/dm/nl/5216/Oosterweel/article/detail/1083110/2010/03/22/Ruim-3-000-Antwerpenaren-zullen-voortijdig-sterven-door-verkeerslawaai.dhtml
ARCHIVED: http://noisetube.net/publications/DeMorgen.be_22-03-2010.pdf

Web Het Laatste Nieuws. *"Ruim 3.000 Antwerpenaren zullen voortijdig sterven door verkeerslawaai"*. Mar. 22, 2010. In Dutch.
URL: http://www.hln.be/hln/nl/957/Belgie/article/detail/1083110/2010/03/22/Ruim-3-000-Antwerpenaren-zullen-voortijdig-sterven-door-verkeerslawaai.dhtml
ARCHIVED: http://noisetube.net/publications/HLN.be_2010-03-22.pdf

Print "Herrie om Lange Wapper: de geheime geluidskaart". In: *Humo* 3629 | 12 (Mar. 23, 2010), pp. 28–29. In Dutch.
ARCHIVED: http://noisetube.net/publications/Humo2010-03-23.pdf

Print "Meer dan 3.000 Antwerpenaren zullen vroegtijdig overlijden door verkeerslawaai". In: *Het Nieuwsblad. Antwerpen* (Mar. 23, 2010), p. 23. In Dutch.
ARCHIVED: http://noisetube.net/publications/HNBApen23-03-2010.pdf

Print Filip Marsboom. "Ademloos meet geluidsoverlast. Merendeel Antwerpenaars leeft in ongezond gebied". In: *Het Laatste Nieuws. Antwerpen* (Mar. 23, 2010), p. 19. In Dutch.
ARCHIVED: http://www.vub.ac.be/infoover/media/uab20100323.html#doc3

Print "VUB meet verkeerslawaai met gsm". In: *Gazet Van Antwerpen* (Mar. 23, 2010), p. 33. In Dutch.
ARCHIVED: http://www.vub.ac.be/infoover/media/uab20100323.html#doc4

Radio BBC World Service. *Digital Planet*. Sept. 21, 2010. Short interview with Matthias Stevens.
URL: http://www.bbc.co.uk/programmes/p009r50h
ARCHIVED: http://noisetube.net/publications/InterviewBBCWorld.mp3

Print Brandon Bailey. “Keeping watch in the environment: Smartphone apps gather information on creeks, birds, hazards”. In: *San Jose Mercury News* (Oct. 18, 2010), pp. 21–22.
ARCHIVED: http://noisetube.net/publications/SanJoseMercuryNews_18-10-2010.pdf

Web Brandon Bailey. *Smartphones used to gather data about San Jose’s urban streams*. MercuryNews.com. Oct. 18, 2010.
URL: http://www.mercurynews.com/ci_16350288 (offline)
ARCHIVED: http://noisetube.net/publications/MercuryNews.com_18-10-2010.pdf

2011

TV 3sat. *nano. Mit dem Handy gegen Lärm und Schmutz, Brüsseler Forscher sammeln Daten per App*. Apr. 8, 2011. Interview with Matthias Stevens and Ellie D’Hondt. In German. Rebroadcast on Deutsche Welle TV.
URL: http://www.3sat.de/mediathek/?mode=play&obj=24317
ARCHIVED: http://noisetube.net/publications/3sat_nano_2011-04-08.mp4

Web 3sat. *nano. Mit dem Handy gegen Lärm und Schmutz, Brüsseler Forscher sammeln Daten per App*. Apr. 8, 2011. In German.
URL: http://3sat.de/?153407
ARCHIVED: http://noisetube.net/publications/3sat_nano_2011-04-08Web.pdf

TV 3sat. *neues. Mobiles Crowdsourcing, Der Mensch als Sensor*. Apr. 10, 2011. Interview with Matthias Stevens and Ellie D’Hondt. In German. Rebroadcast on Deutsche Welle TV.
URL: http://www.3sat.de/mediathek/?mode=play&obj=24312 (offline)
ARCHIVED: http://noisetube.net/publications/3sat_neues_2011-04-10.mp4

Web 3sat. *neues. Mobiles Crowdsourcing, Der Mensch als Sensor*. Apr. 10, 2011. In German.
URL: http://3sat.de/?153433
ARCHIVED: http://noisetube.net/publications/3sat_neues_2011-04-10Web.pdf

Print “Can ICT save the planet? ICT: part of the problem as well as the solution”. Round table. In: *One* 5.13 (Q4 2011). English edition, pp. 28–32.
URL: http://www.onemagazine.be/pdf/one-13/ONE13_Q4_EN_2011.pdf
ARCHIVED: http://noisetube.net/publications/One2011Q4EN.pdf

Print "Kan ICT de planeet redden? ICT: deel van het probleem én van de oplossing". Round table. In: *One* 5.13 (Q4 2011). Dutch edition, pp. 28–32.
URL: http://www.onemagazine.be/pdf/one-13/ONE13_Q4_NL_2011.pdf
ARCHIVED: http://noisetube.net/publications/One2011Q4NL.pdf

Print "L'ICT peut-il sauver la planète ? L'ICT : part du problème et de la solution". Round table. In: *One* 5.13 (Q4 2011). French edition, pp. 28–32.
URL: http://www.onemagazine.be/pdf/one-13/ONE13_Q4_FR_2011.pdf
ARCHIVED: http://noisetube.net/publications/One2011Q4FR.pdf

Web One. *Can ICT save the planet? ICT: part of the problem as well as the solution*. Round table. Oct. 12, 2011. English edition.
URL: http://www.onemagazine.be/2011/10/12/can-ict-save-the-planet
ARCHIVED: http://noisetube.net/publications/One2011Q4WebEN.pdf

Web One. *Kan ICT de planeet redden? ICT: deel van het probleem én van de oplossing*. Round table. Oct. 12, 2011. Dutch edition.
URL: http://www.onemagazine.be/nl/2011/10/12/can-ict-save-the-planet/
ARCHIVED: http://noisetube.net/publications/One2011Q4WebNL.pdf

Web One. *L'ICT peut-il sauver la planète ? L'ICT : part du problème et de la solution*. Round table. Oct. 12, 2011. French edition.
URL: http://www.onemagazine.be/fr/2011/10/12/can-ict-save-the-planet/
ARCHIVED: http://noisetube.net/publications/One2011Q4WebFR.pdf

2012

Web Lauren Hockenson. *How One App Empowers Citizens to Take Control of Urban Noise Pollution*. Mashable. Jan. 11, 2012.
URL: http://mashable.com/2012/01/11/noisetube-noise-pollution
ARCHIVED: http://noisetube.net/publications/Mashable.com_2012-01-11.pdf

Print Tuur De Moor. "Op avontuur door het geluidslandschap van de stad". In: *Agenda* 1312 (Jan. 20, 2012), p. 27. In Dutch.
URL: http://www.bdw.be/nl/agenda/magazine/2012-01-20
ARCHIVED: http://noisetube.net/publications/Agenda_2012-01-20.pdf

Print Geert Van der Speeten. "Brusselse geluidsoverlast meter met je gsm. NoiseTube brengt probleemgebieden in kaart". In: *De Standaard* 89.19 (Jan. 23, 2012), p. D3. In Dutch.
ARCHIVED: http://www.vub.ac.be/infoover/media/uab20120123.html#doc10

Web Geert Van der Speeten. *Geluidskaarten voor Brussel. NoiseTube brengt probleemgebieden in kaart*. De Standaard. Jan. 23, 2012. In Dutch.
URL: http://www.standaard.be/mobilia/cnt/J73L7HRC
Archived: http://noisetube.net/publications/DS.be_2012-01-23.pdf

Web Brusselnieuws.be. *Geluidsoverlast in kaart brengen met smartphone*. Jan. 23, 2012. In Dutch.
URL: http://www.brusselnieuws.be/artikel/geluidsoverlast-kaart-brengen-met-smartphone
Archived: http://noisetube.net/publications/BrusselNieuws_2012-01-23.pdf

TV Télé Bruxelles. *Semaine du son: rallye sonore et mesures sonores*. Jan. 23, 2012. Interview with Ellie D'Hondt. In French.
URL: http://www.telebruxelles.net/portail/info/info-culturelle/17642-semaine-du-son-rallye-sonore-et-mesures-sonores
Archived: http://noisetube.net/publications/TeleBruxelles_2012-01-23.mp4

Radio FM Brussel. *Flagey 18*. Jan. 24, 2012. Interview with Ellie D'Hondt. In Dutch.
Archived: http://noisetube.net/publications/FMBrussel_2012-01-24.mp3

Print Maxie Eckert. "'We staan er niet bij stil hoeveel lawaai we verdragen'. Het geluid van Brussel in kaart gebracht". Reportage. In: *De Standaard* 89.21 (Jan. 25, 2012), p. 10. In Dutch.
Archived: http://www.vub.ac.be/infoover/media/uab20120125.html#doc3

Web Maxie Eckert. *'We staan er niet bij stil hoeveel lawaai we verdragen'. Het geluid van Brussel in kaart gebracht*. Reportage. De Standaard. Jan. 25, 2012. In Dutch.
URL: http://www.standaard.be/artikel/detail.aspx?artikelid=M03LAN48
Archived: http://noisetube.net/publications/DS.be_2012-01-25.pdf

TV TV Brussel. *Brusselse geluidsoverlast meten met je smartphone*. Jan. 26, 2012. Interview with Ellie D'Hondt. In Dutch.
URL: http://tvbrussel.be/video/4/brusselse-geluidsoverlast-meten-met-je-smartphone
Archived: http://noisetube.net/publications/TVBrussel_2012-01-26.mp4

Print Christian Du Brulle. "Le bruit pisté par les Bruxellois. Une chercheuse traque les nuisances sonores via les smartphones". Environnement. In: *Le Soir. Bruxelles/Bruxelles Périphérie/Brabant-Wallon* 25 (Jan. 30, 2012), p. 28. In French.
Archived: http://noisetube.net/publications/LeSoir_2012-01-30.pdf

Web Christian Du Brulle. *Le bruit pisté par les Bruxellois*. Le Soir. Jan. 30, 2012. In French.
URL: http://lesoir.be/regions/bruxelles/2012-01-30/le-bruit-piste-par-les-bruxellois-893703.php
Archived: http://noisetube.net/publications/LeSoir_2012-01-30Web.pdf

Radio Radio Contact. *Le 16H - 20H*. Jan. 30, 2012. Interview with Ellie D'Hondt. In French.
URL: http://www.radiocontact.be/page/radio-contact-les-emissions-le-16h-20h/599.aspx

Radio Bel RTL. *RTL Régions*. Feb. 21, 2012. Interview with Matthias Stevens. In French.
URL: http://www.rtl.be/videobelrtl/video/383958.aspx?CategoryID=2368
Archived: http://noisetube.net/publications/BelRTL_RTLRegions_2012-02-21.mp3

Web David Larousserie. *Les profanes jouent avec les experts*. Le Monde. Mar. 2, 2012. In French.
URL: http://lemonde.fr/sciences/article/2012/03/02/les-profanes-jouent-avec-les-experts_1650607_1650684.html
Archived: http://noisetube.net/publications/LeMonde.fr2012-03-02.pdf

Print David Larousserie. "Les sciences citoyennes. Les profanes jouent avec les experts". Innovation. In: *Le Monde* 68.20876 (Mar. 3, 2012): *Science & Techno*, pp. 4–5. In French.
Archived: http://noisetube.net/publications/LeMonde2012-03-03.pdf

Web Gazet Van Antwerpen. *Tuinwijk meet zelf lawaai snelweg*. Borgerhout. Mar. 26, 2012. In Dutch.
URL: http://www.gva.be/regio-antwerpen-stad/borgerhout/tuinwijk-meet-zelf-lawaai-snelweg.aspx
Archived: http://noisetube.net/publications/GVA.be2012-03-26.pdf

Print "Tuinwijk gaat zelf lawaai snelweg meten". Borgerhout. In: *Gazet Van Antwerpen. Metropool Noord* (Mar. 26, 2012), p. 59. In Dutch.
Archived: http://www.vub.ac.be/infoover/media/uab20120326.html#doc4

Print "Bewoners Tuinwijk gaan zelf lawaai van snelweg meten". Borgerhout. In: *Gazet Van Antwerpen. Metropool Zuid* (Mar. 26, 2012), p. 59. In Dutch.
Archived: http://www.vub.ac.be/infoover/media/uab20120326.html#doc5

Print Marion Orlans. "NoiseTube, de app die jacht maakt op lawaai". In: *My Sphere* (Apr. 1, 2012). Dutch edition, p. 10.
Archived: http://noisetube.net/publications/MySphere2012-04-01NL.pdf

Print Marion Orlans. "NoiseTube, l'appli qui traque les bruits". In: *My Sphere* (Apr. 1, 2012). French edition, p. 10.
Archived: http://noisetube.net/publications/MySphere2012-04-01FR.pdf

G.4 NoiseTube campaigns

Perhaps the most effective dissemination efforts are the participatory noise mapping campaigns we have organised in collaboration with (mostly) volunteers. Here is an overview:

- Antwerp, Belgium:
  - July & November 2010: campaigns at Linkeroever with volunteers of *Ademloos* (see chapter 7);
  - March 2012: campaign at Tuinwijk with volunteers of the *Notenkrakers*;
- Brussels, Belgium:
  - December 2010: on and around VUB campus Etterbeek, with 45 students;
  - January 29, 2011: noise mapping walk during La Semaine du Son;
  - January 1-29, 2012: noise mapping contest for La Semaine du Son;
  - Spring 2012: BruSense project campaigns (see section 8.4);
- San Francisco, USA, June 2011: demonstration campaign for Pervasive 2011 [126].

Bibliography

[1] David M. Aanensen, Derek M. Huntley, Edward J. Feil, Fada'a al-Own, and Brian G. Spratt. "EpiCollect: Linking Smartphones to Web Applications for Epidemiology, Ecology and Community Data Collection". In: *PLoS ONE* 4.9 (Sept. 2009), e6968.
ISSN: 1932-6203
DOI: 10.1371/journal.pone.0006968

[2] Tarek Abdelzaher et al. "Mobiscopes for Human Spaces". In: *IEEE Pervasive Computing* 6.2 (Apr. 2007). Special issue on Building a Sensor-Rich World, pp. 20–29.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2007.38

[3] ABI Research. *Samsung's Q2 Smartphone Growth Shows that the Race Is Its to Lose*. Press release. July 28, 2011.
URL: http://www.abiresearch.com/press/3740

[4] Acoustical Society of America. "American Tentative Standards for Sound Level Meters for Measurement of Noise and Other Sounds". In: *Journal of the Acoustical Society of America* 8 (1936), pp. 147–152. Adopted as ANSI/ASA Z24.3-1936.
ISSN: 0001-4966

[5] Ademloos vzw. *Ademloos*. In Dutch.
URL: http://www.ademloos.be

[6] AFP. *Campaigners fight noise pollution in bustling Mumbai*. Video posted on YouTube. Mar. 2009.
URL: http://www.youtube.com/watch?v=-zxNVWI2kNE

[7] E. Agapie et al. "Seeing our signals: combining location traces and web-based models for personal discovery". In: *HotMobile '08 – Proceedings of the 9th workshop on Mobile computing systems and applications (Napa, CA, USA, February 25-26, 2008)*. New York, NY, USA: ACM, pp. 6–10.
ISBN: 978-1-60558-118-7
DOI: 10.1145/1411759.1411762

[8] Agilent Technologies. *33120A Function / Arbitrary Waveform Generator*. Formerly manufactured by HP.
URL: http://www.home.agilent.com/agilent/product.jspx?pn=33120A

[9] Christophe Aguiton and Dominique Cardon. "The Strength of Weak Cooperation: An Attempt to Understand the Meaning of Web 2.0". In: *Communications & Strategies* 65 (First Quarter 2007): *Web 2.0 the Internet as a digital common*, pp. 51–65.
ISSN: 1157-8637
URL: http://idate.org/en/Digiworld-store/Collection/Communications-Strategies_18/No-65-Web-2-0-the-Internet-as-a-digital-common_434.html

[10] Alcatel-Lucent Bell NV. *SenseTale*.
URL: http://sensetale.com

[11] Myles R. Allen. "Do-it-yourself climate prediction". Commentary. In: *Nature* 401.6754 (Oct. 1999), p. 642.
ISSN: 0028-0836
DOI: 10.1038/44266

[12] Myles R. Allen. "Climate forecasting: Possible or probable?" Concepts. In: *Nature* 425.6955 (Sept. 2003), p. 242.
ISSN: 0028-0836
DOI: 10.1038/425242a

[13] Amazon.com, Inc. *Mechanical Turk*.
URL: https://www.mturk.com

[14] AmbientTalk Team. *AmbientTalk*. Vrije Universiteit Brussel, Software Languages Lab.
URL: http://soft.vub.ac.be/amop

[15] American National Standards Institute. *American National Standard Specification for Sound Level Meters*. ANSI S1.4-1983 (R2006)/ANSI S1.4A-1985 (R2006). 1983-2006.
URL: http://webstore.ansi.org/RecordDetail.aspx?sku=ANSI+S1.4-1983+(R2006)%2fANSI+S1.4a-1985+(R2006)

[16] American National Standards Institute. *American National Standard Specification for Personal Noise Dosimeters*. ANSI S1.25-1991 (R2007). 1991-2007.
URL: http://webstore.ansi.org/RecordDetail.aspx?sku=ANSI+S1.25-1991+(R2007)

[17] American National Standards Institute. *Specifications for Integrating-Averaging Sound Level Meters*. ANSI ANSI S1.43-1997 (R2007). 1997-2007.
URL: http://webstore.ansi.org/RecordDetail.aspx?sku=ANSI+S1.43-1997+(R2007)

[18] David P. Anderson. "BOINC: A System for Public-Resource Computing and Storage". In: *Fifth IEEE/ACM International Workshop on Grid Computing (8 November 2004, Pittsburgh, USA). Proceedings*, pp. 4–10.
ISSN: 1550-5510
DOI: 10.1109/GRID.2004.14

[19] David P. Anderson, Jeff Cobb, Eric Korpela, Matt Lebofsky, and Dan Werthimer. "SETI@home: An Experiment in Public-Resource Computing". In: *Communications of the ACM* 45.11 (Nov. 2002), pp. 56–61.
ISSN: 0001-0782
DOI: 10.1145/581571.581573

[20] Robert Anderson, Branislav Mikuliç, Greet Vermeylen, Maija Lyly-Yrjänäinen, and Valentina Zigante. *Second European Quality of Life Survey – Overview*. Report ef0902. European Foundation for the Improvement of Living and Working Conditions (Eurofound), Mar. 2007. 108 pp. Survey results can be explored online [172].
ISBN: 978-92-897-0847-0
URL: http://www.eurofound.europa.eu/surveys/eqls/2007

[21] Paul M. Aoki et al. "Common Sense: Mobile Environmental Sensing Platforms to Support Community Action and Citizen Science". In: *Adjunct Proceedings of the 10th International Conference on Ubiquitous Computing (Ubicomp 2008)*, pp. 59–60.
URL: http://www.paulaoki.com/papers/ubicomp08-demo.pdf

[22] Paul M. Aoki et al. "A Vehicle for Research: Using Street Sweepers to Explore the Landscape of Environmental Community Action". In: *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems (April 4-9, 2009, Boston, MA, USA)*. New York, NY, USA: ACM, pp. 375–384.
ISBN: 978-1-60558-246-7
DOI: 10.1145/1518701.1518762

[23] Apache Software Foundation. *Apache Harmony: Class Library Component Status (API completeness)*.
URL: http://harmony.apache.org/subcomponents/classlibrary/status.html

[24] Apache Software Foundation. *Apache Harmony project*.
URL: http://harmony.apache.org

[25] Apache Software Foundation. *Apache HttpComponents project: HttpClient*.
URL: http://hc.apache.org/httpcomponents-client-ga

[26] Apache Software Foundation. *Apache HTTP Server*. Sometimes referred to as *Apache httpd*, or just *Apache*.
URL: http://httpd.apache.org

[27] Apache Software Foundation. *Apache Module mod_rewrite*.
URL: http://httpd.apache.org/docs/current/mod/mod_rewrite.html

[28] Apache Software Foundation. *Apache License, Version 2.0*. 2004.
URL: http://www.apache.org/licenses/LICENSE-2.0.html

[29] Apple, Inc. *iPhone*.
URL: http://www.apple.com/iphone

[30] Apple, Inc. *iOS*. Formerly known as *iPhone OS*.
URL: http://www.apple.com/ios

[31] Apple, Inc. *iTunes App Store*.
URL: http://www.itunes.com/appstore

[32] Apple, Inc. *Apple's App Store Downloads Top 15 Billion*. Press release. July 7, 2011.
URL: http://www.apple.com/pr/library/2011/07/07Apples-App-Store-Downloads-Top-15-Billion.html

[33] Armchair Applications / jjbunn. *AudioTool*. Android app.
URL: https://market.android.com/details?id=com.julian.apps.AudioTool

[34] Audacity Team. *Audacity*. 2000–2012.
URL: http://audacity.sourceforge.net

[35] Audioboo Ltd. *audioBoo*.
URL: http://audioboo.fm

[36] *Awaaz Foundation*. NGO based in Mumbai, India.
URL: http://www.awaaz.org

[37] Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. *Four Degrees of Separation*. Preprint article. Nov. 19, 2011.
URL: http://arxiv.org/abs/1111.4570

[38] Jesse R. Barber, Kevin R. Crooks, and Kurt M. Fristrup. "The costs of chronic noise exposure for terrestrial organisms". Review. In: *Trends in Ecology & Evolution* 25.3 (Mar. 2010), pp. 180–189.
ISSN: 0169-5347
DOI: 10.1016/j.tree.2009.08.002

[39] Sander Bartholomees. "Context-Aware Urban Sensing". Promoted by Wolfgang De Meuter. Advised by Matthias Stevens and Ellie D'Hondt. MA thesis. Vrije Universiteit Brussel, Faculteit Wetenschappen en Bio-ingenieurswetenschappen, Vakgroep Computerwetenschappen, June 2011. 128 pp.
URL: http://brussense.be/Pubs/files/Bartholomees2011.pdf

[40] BBC News. *Facebook driving mobile net usage*. July 14, 2009.
URL: http://news.bbc.co.uk/2/hi/technology/8149652.stm

[41] Matt Walker. *Noise pollution threatens animals*. BBC News. Oct. 14, 2009.
URL: http://news.bbc.co.uk/earth/hi/earth_news/newsid_8305000/8305320.stm

[42] Rory Cellan-Jones. *Hackers and hippies: The origins of social networking*. BBC News. Jan. 25, 2011.
URL: http://www.bbc.co.uk/news/technology-12224588

[43] BBC News. *Legal row over Carrier IQ 'surveillance' app claims*. Dec. 1, 2011.
URL: http://www.bbc.co.uk/news/technology-15982225

[44] Peter Beaumont. *The truth about Twitter, Facebook and the uprisings in the Arab world*. The Guardian. Feb. 25, 2011.
URL: http://www.guardian.co.uk/world/2011/feb/25/twitter-facebook-uprisings-arab-libya

[45] Luca Becchetti, Luca Filipponi, and Andrea Vitaletti. "Opportunistic privacy preserving monitoring". In: *PhoneSense 2010: International Workshop on Sensing for App Phones (November 2, 2010), held at ACM SenSys '10 (Zurich, Switzerland, November 2-5, 2010)*, pp. 51–55.
URL: http://sensorlab.cs.dartmouth.edu/phonesense/program.html

[46] Richard A. Becker et al. "A Tale of One City: Using Cellular Network Data for Urban Planning". In: *IEEE Pervasive Computing* 10.4 (Oct. 2011): *Large-scale opportunistic sensing*, pp. 18–26.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2011.44

[47] Paolo Bellavista, Axel Küpper, and Sumi Helal. "Location-Based Services: Back to the Future". In: *IEEE Pervasive Computing* 7.2 (Apr. 2008), pp. 85–89.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2008.34

[48] Yochai Benkler. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. Yale University Press, Apr. 2006. 528 pp.
ISBN: 978-0300110562
URL: http://yalepress.yale.edu/yupbooks/book.asp?isbn=9780300110562

[49] Yochai Benkler and Helen Nissenbaum. "Commons-based Peer Production and Virtue". In: *Journal of Political Philosophy* 14.4 (Dec. 2006), pp. 394–419.
ISSN: 1467-9760
DOI: 10.1111/j.1467-9760.2006.00235.x

[50] Elliott H. Berger, Larry H. Royster, Julia D. Royster, Dennis P. Driscoll, and Marty Layne, eds. *The Noise Manual*. Revised Fifth Edition. American Industrial Hygiene Association (AIHA), Aug. 2003. 796 pp.
ISBN: 978-1931504027
URL: http://www.aiha.org/insideaiha/volunteergroups/Noise

[51] Karin Bijsterveld. *Mechanical Sound: Technology, Culture, and Public Problems of Noise in the Twentieth Century*. Inside Technology. Cambridge, MA, USA: MIT Press, Aug. 2008. 368 pp.
ISBN: 978-0-262-02639-0
URL: http://mitpress.mit.edu/0262026392

[52] Dan Bornstein et al. *Dalvik Virtual Machine*. Google, Inc.
URL: http://code.google.com/p/dalvik

[53] Gaetano Borriello, Matthew Chalmers, Anthony LaMarca, and Paddy Nixon. "Delivering Real-World Ubiquitous Location Systems". In: *Communications of the ACM* 48.3 (Mar. 2005). Special issue on "The disappearing computer", pp. 36–41.
ISSN: 0001-0782
DOI: 10.1145/1047671.1047701

[54] Raja Bose. "Sensor Networks—Motes, Smart Spaces, and Beyond". In: *IEEE Pervasive Computing* 8.3 (July 2009), pp. 84–90.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2009.55

[55] Danah M. Boyd and Nicole B. Ellison. "Social Network Sites: Definition, History, and Scholarship". In: *Journal of Computer-Mediated Communication* 13.1 (Oct. 2007), pp. 210–230.
ISSN: 1083-6101
DOI: 10.1111/j.1083-6101.2007.00393.x

[56] Patrick Brady. *Android Anatomy and Physiology*. Presented at: *Google I/O, May 28-29, 2008, San Francisco, CA, USA*. 2008.
URL: https://sites.google.com/site/io/anatomy--physiology-of-an-android

[57] British Trust for Ornithology. *BTO Volunteer surveys*.
URL: http://www.bto.org/volunteer-surveys

[58] Bruitparif – Observatoire du bruit en Île-de-France. *Déploiement du réseau de surveillance RUMEUR (Réseau Urbain de Mesure de l'Environnement sonore d'Utilité Régionale)*. In French.
URL: http://bruitparif.fr/bruitparif/our-projects/mesurer/deploiement-du-reseau-de-surveillance-rumeur

[59] Bruitparif – Observatoire du bruit en Île-de-France. *Plateform Rumeur*. Web portal for the consultation of data collected via the RUMEUR sensor network. In French.
URL: http://rumeur.bruitparif.fr

[60] Bruitparif – Observatoire du bruit en Île-de-France. *Ouverture de l'espace cartographique*. Official END-compliant noise maps for Paris and the Île-de-France region. In French. 2006–2010.
URL: http://www.bruitparif.fr/ouverture-de-l-espace-cartographique

[61] A.J. Bernheim Brush, John Krumm, and James Scott. "Exploring End User Preferences for Location Obfuscation, Location-Based Services, and the Value of Location". In: *Ubicomp '10 – Proceedings of the 12th ACM international conference on Ubiquitous computing (Copenhagen, Denmark, September 26-29, 2010)*. New York, NY, USA: ACM, pp. 95–104.
ISBN: 978-1-60558-843-8
DOI: 10.1145/1864349.1864381

[62] Brussels Instituut voor Milieubeheer. *Webnoise*. In Dutch or French.
URL: http://app.bruxellesenvironnement.be:8080/WebNoise/Home?lang=nl&langtype=2067

[63] Brussels Instituut voor Milieubeheer. *Netwerk van geluidsmeetstations in het Brussels Hoofdstedelijk Gewest*. Ed. by Catherine Lecointre and Georges Dellisse. De BIM gegevens: "Lawaai in Brussel" 5. In Dutch. 2008. 5 pp.
URL: http://documentatie.leefmilieubrussel.be/documents/Geluid_5_2008.PDF?langtype=2067

[64] Brussels Instituut voor Milieubeheer. *Geluidshinder door het verkeer – Strategische kaart voor het Brussels Hoofdstedelijk Gewest / Bruit des transports – Cartographie stratégique en Région de Bruxelles-Capitale*. Ed. by Marie Poupé, Jean-Laurent Simons, and Marc Magain. In Dutch and French. 2010. 60 pp.
URL: http://www.leefmilieubrussel.be/Templates/Particuliers/Niveau2.aspx?id=4192

[65] BrusSense Team. *Interactive noise maps based on measuring campaign in Linkeroever (Antwerp, Belgium)*. 2010–2011.
URL: http://brussense.be/experiments/linkeroever

[66] BrusSense Team. *BrusSense: Participatory sensing for sustainable urban living*. 2010–2012.
URL: http://brussense.be

[67] BrusSense Team. *Interactive pollution and microclimate maps based on measuring campaigns in Brussels*. 2012.
URL: http://brussense.be/experiments/bxlsense

[68] Susan L. Bryant, Andrea Forte, and Amy Bruckman. “Becoming Wikipedian: transformation of participation in a collaborative online encyclopedia”. In: *GROUP '05: Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work (November 6-9, 2005, Sanibel Island, FL, USA)*. New York, NY, USA: ACM, pp. 1–10.
DOI: 10.1145/1099203.1099205

[69] Emma Buckland. *Smartphones will outnumber other handsets by 2014 in Western Europe*. Analysys Mason. Aug. 11, 2011.
URL: http://www.analysysmason.com/About-Us/News/Insight/Smartphone_penetration_Aug2011

[70] Leah Buechley, Eric Paulos, Daniela Rosner, and Amanda Williams, eds. *DIY::HCI. Proceedings of the “DIY for CHI” workshop (April 5, 2009), held at CHI '09, the 27th International Conference on Human Factors in Computing Systems (April 4-9, 2009, Boston, MA, USA)*. 160 pp.
URL: http://people.ischool.berkeley.edu/~daniela/diy

[71] Jeffrey A. Burke et al. “Participatory Sensing”. In: *WSW'06: Workshop on World-Sensor-Web (October 31, 2006), held at ACM SenSys '06 (October 31-November 3, 2006, Boulder, CO, USA)*.
URL: http://www.sensorplanet.org/wsw2006/6_Burke_wsw06_ucla_final.pdf

[72] Cambridge University Press. *Cambridge Dictionaries Online – Cambridge Advanced Learner's Dictionary & Thesaurus*.
URL: http://dictionary.cambridge.org/dictionary/british

[73] Alison Campbell et al. *Review of the Literature on the Links between Biodiversity and Climate Change: Impacts, Adaptation and Mitigation*. United Nations Environment Programme, World Conservation Monitoring Centre (UNEP-WCMC). CBD Technical Series 42. Montreal, Quebec, Canada: Secretariat of the Convention on Biological Diversity, Jan. 2010. 124 pp.
ISBN: 92-9225-136-8
URL: http://www.unep-wcmc.org/review-of-the-literature-on-the-links-between-biodiversity-and-climate-change_155.html

[74] Andrew T. Campbell, Shane B. Eisenman, Nicholas D. Lane, Emiliano Miluzzo, and Ronald A. Peterson. “People-Centric Urban Sensing”. In: *WICON '06: Proceedings of the 2nd annual international workshop on Wireless internet (August 2-5, 2006, Boston, MA, USA)*. Invited paper. New York, NY, USA: ACM, p. 18.
ISBN: 1-59593-510-X
DOI: 10.1145/1234161.1234179

[75] Andrew T. Campbell et al. “The Rise of People-Centric Sensing”. In: *IEEE Internet Computing* 12.4 (July 2008). Special Issue on Mesh Networks, pp. 12–21.
ISSN: 1089-7801
DOI: 10.1109/MIC.2008.90

[76] Canalys. *Smart phones defy slowdown – Nokia retains lead, with Apple and RIM rising fast, but Microsoft loses ground.* Press release. Aug. 17, 2009.
URL: http://www.canalys.com/pr/2009/r2009081.html

[77] Canalys. *Global smart phone market growth rises to 67% – Nokia and RIM achieve record Q1 volumes.* Press release. May 3, 2010.
URL: http://www.canalys.com/pr/2010/r2010051.html

[78] Ciro Cattuto, Alain Barrat, Andrea Baldassarri, Gregory Schehr, and Vittorio Loreto. "Collective dynamics of social annotation". In: *Proceedings of the National Academy of Sciences (PNAS)* 106.26 (June 2009), pp. 10511–10515.
ISSN: 0027-8424
DOI: 10.1073/pnas.0901136106

[79] Ciro Cattuto, Vittorio Loreto, and Luciano Pietronero. "Semiotic dynamics and collaborative tagging". In: *Proceedings of the National Academy of Sciences (PNAS)* 104.5 (Jan. 2007), pp. 1461–1464.
ISSN: 0027-8424
DOI: 10.1073/pnas.0610487104

[80] Ciro Cattuto et al. "Network properties of folksonomies". In: *AI Communications* 20.4 (Nov. 2007), pp. 245–262.
ISSN: 0921-7126
URL: http://iospress.metapress.com/content/8x34022x43427k3h

[81] Center for Embedded Networked Sensing, UCLA. *Biketastic.*
URL: http://biketastic.com

[82] Center for Embedded Networked Sensing, UCLA. *Campaignr.*
URL: http://wiki.urban.cens.ucla.edu/index.php/Campaignr

[83] Center for Embedded Networked Sensing, UCLA. *CycleSense project.*
URL: http://urban.cens.ucla.edu/projects/cyclesense

[84] Center for Embedded Networked Sensing, UCLA. *PEIR, the Personal Environmental Impact Report.*
URL: http://urban.cens.ucla.edu/projects/peir

[85] Center for Embedded Networked Sensing, UCLA. *WhatsNoisy.*
URL: http://whatsnoisy.appspot.com

[86] CERN, UNITAR, and University of Geneva. *Citizen Cyberscience Centre.*
URL: http://www.citizencyberscience.net

[87] Alok Choudhary, William Hendrix, Kathy Lee, Diana Palsetia, and Wei-Keng Liao. "Social Media Evolution of the Egyptian Revolution". In: *Communications of the ACM* 55.5 (May 2012), pp. 74–80.
ISSN: 0001-0782
DOI: 10.1145/2160718.2160736

[88] Tanzeem Choudhury et al. "The Mobile Sensing Platform: An Embedded Activity Recognition System". In: *IEEE Pervasive Computing* 7.2 (Apr. 2008), pp. 32–41.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2008.39

[89] Carl Christensen, Tolu Aina, and David Stainforth. "The challenge of volunteer computing with lengthy climate model simulations". In: *First International Conference on e-Science and Grid Computing (Melbourne, Australia, 5-8 December, 2005)*. Ed. by Heinz Stockinger, Rajkumar Buyya, and Ron Perrott. Washington, DC, USA: IEEE Computer Society, pp. 8–15.
ISBN: 0-7695-2448-6
DOI: 10.1109/E-SCIENCE.2005.76

[90] Cirrus Environmental. *An Introduction to Environmental Noise Parameters*.
URL: http://cirrusenvironmental.co.uk/links_and_information/environmental_noise_descriptors/environmental_noise_descriptors_page_1.php

[91] Steve Coast et al. *CloudMade*.
URL: http://cloudmade.com

[92] Jeffrey P. Cohn. "Citizen Science: Can Volunteers Do Real Research?" In: *BioScience* 58.3 (Mar. 2008), pp. 192–197.
ISSN: 0006-3568
DOI: 10.1641/BS80303

[93] Colorado Climate Center. *Community Collaborative Rain, Hail & Snow Network (CoCoRaHS)*. 1997.
URL: http://www.cocorahs.org

[94] Ken Colstad and Efrem Lipkin. "Community Memory: A Public Information Network". In: *ACM SIGCAS Computers and Society* 6.4 (Winter 1975), pp. 6–7.
ISSN: 0095-2737
DOI: 10.1145/958785.958788

[95] Committee on Technology for a Quieter America, National Academy of Engineering. *Technology for a Quieter America*. National Academies Press, 2010. 216 pp.
ISBN: 978-0-309-15632-5
URL: http://www.nap.edu/catalog.php?record_id=12928

[96] Marisol Concha-Barrientos, Diarmid Campbell-Lendrum, and Kyle Steenland. *Occupational noise: Assessing the burden of disease from work-related hearing impairment at national and local levels*. WHO Environmental Burden of Disease Series 9. Geneva: World Health Organization, 2004. 41 pp.
ISBN: 92-4-159192-7
ISSN: 1728-1652
URL: http://www.who.int/quantifying_ehimpacts/national/en

[97] Congress of the United States of America. "Noise Control Act of 1972, Public Law 92-574". In: *United States Statutes at Large* 86 (Oct. 27, 1972): *92nd Congress, 2nd Session*, pp. 1234–1250.
URL: http://www.gpo.gov/fdsys/search/pagedetails.action?packageId=STATUTE-086

[98] Sunny Consolvo et al. "Activity Sensing in the Wild: A Field Trial of UbiFit Garden". In: *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (Florence, Italy, April 5-10, 2008)*. New York, NY, USA: ACM, pp. 1797–1806.
ISBN: 978-1-60558-011-1
DOI: 10.1145/1357054.1357335

[99] Seth Cooper et al. "Predicting protein structures with a multiplayer online game". Letter. In: *Nature* 466.7307 (Aug. 2010), pp. 756–760.
ISSN: 0028-0836
DOI: 10.1038/nature09304

[100] Jon Corbett et al. "Overview: Mapping for Change – the emergence of a new practice". In: *Mapping for change: Practice, technologies and communication*. Ed. by Giacomo Rambaldi et al. Participatory Learning and Action 54. International Institute for Environment, Development (IIED), London, UK / Technical Centre for Agricultural, and Rural Development (CTA), Wageningen, The Netherlands, Apr. 2006. Chap. 1, pp. 13–19.
ISBN: 978-1-84369-605-6
ISSN: 1357-938X
URL: http://pubs.iied.org/14507IIED.html

[101] Jason Corburn. "Confronting the Challenges in Reconnecting Urban Planning and Public Health". In: *American Journal of Public Health* 94.4 (Apr. 2004), pp. 541–546.
ISSN: 0090-0036
DOI: 10.2105/AJPH.94.4.541

[102] Jason Corburn. *Street Science: Community Knowledge and Environmental Health Justice*. MIT Press, Aug. 2005. 284 pp.
ISBN: 978-0-262-53272-3
URL: http://mitpress.mit.edu/9780262532723

[103] Christophe Couvreur. *Octave toolbox for MATLAB*. Generates normalised A-weigthing, C-weighting, octave and one-third-octave digital filters. Faculte Polytechnique de Mons. Dec. 29, 1997.
URL: http://www.mathworks.com/matlabcentral/fileexchange/69

[104] Douglas Crockford. *JSON (JavaScript Object Notation)*. Standardised as IETF RFC 4627.
URL: http://json.org

[105] Eduardo Cuervo et al. "MAUI: Making Smartphones Last Longer with Code Offload". In: *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services (San Francisco, CA, USA, June 15-18, 2010)*. New York, NY, USA: ACM, pp. 49–62.
ISBN: 978-1-60558-985-5
DOI: 10.1145/1814433.1814441

[106] Dana Cuff, Mark Hansen, and Jerry Kang. "Urban Sensing: Out of the Woods". In: *Communications of the ACM* 51.3 (Mar. 2008), pp. 24–33.
ISSN: 0001-0782
DOI: 10.1145/1325555.1325562

[107] David Culler, Deborah Estrin, and Mani Srivastava. "Overview of Sensor Networks". Guest Editors' Introduction. In: *Computer* 37.8 (Aug. 2004), pp. 41–49.
ISSN: 0018-9162
DOI: 10.1109/MC.2004.93

[108] Thomas E. Curry, Stephen Ansolabehere, and Howard Herzog. *A Survey of Public Attitudes towards Climate Change and Climate Change Mitigation Technologies in the United States: Analyses of 2006 Results*. MIT LFEE 2007-01 WP. Massachusetts Institute of Technology, Laboratory for Energy and the Environment, Apr. 2007. 48 pp.
URL: http://sequestration.mit.edu/bibliography/policy.html

[109] Jeffrey L. Danhauer et al. "Survey of College Students on iPod Use and Hearing Health". In: *Journal of the American Academy of Audiology* 20.1 (Jan. 2009), pp. 5–27.
ISSN: 1050-0545
DOI: 10.3766/jaaa.20.1.2

[110] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of Wireless Sensor Networks: Theory and Practice*. Wireless Communications and Mobile Computing. Wiley, July 2010. 330 pp.
ISBN: 978-0-470-99765-9
URL: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470997656.html

[111] Dartmouth College. *Smartphone Sensing Group*. Previously known as the *Mobile Sensing Group*.
URL: http://sensorlab.cs.dartmouth.edu

[112] Tathagata Das, Prashanth Mohan, Venkata N. Padmanabhan, Ramachandran Ramjee, and Asankhaya Sharma. "PRISM: Platform for Remote Sensing using Smartphones". In: *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services (San Francisco, CA, USA, June 15-18, 2010)*. New York, NY, USA: ACM, pp. 63–76.
ISBN: 978-1-60558-985-5
DOI: 10.1145/1814433.1814442

[113] Debian Project. *Debian GNU/Linux*.
URL: http://www.debian.org

[114] Nico Deblauwe. "GSM-based Positioning: Techniques and Application". PhD thesis. Vrije Universiteit Brussel, Faculty of Engineering, Department of Fundamental Electricity and Instrumentation, 2008.
URL: http://homepages.vub.ac.be/~ndeblauw

[115] Jessie Dedecker. "Ambient-Oriented Programming". PhD thesis. Vrije Universiteit Brussel, Faculteit Wetenschappen, Vakgroep Informatica, May 2006. 325 pp.
URL: http://prog.vub.ac.be/Publications/2006/vub-prog-phd-06-01.pdf

[116] Jessie Dedecker, Tom Van Cutsem, Stijn Mostinckx, Theo D'Hondt, and Wolfgang De Meuter. "Ambient-oriented programming". In: *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications (OOPSLA '05; San Diego, CA, USA, October 16-20, 2005)*. New York, NY, USA: ACM, pp. 31–40.
ISBN: 1-59593-193-7
DOI: 10.1145/1094855.1094867

[117] Boris Defréville, Pierre Philippe, Catherine Lavandier, Pierre Roy, and Yann Françoise. "Objective Representation of Urban Soundscape: Application to a Parisian Neighbourhood". In: *Marseille'06: 3rd Southern Europe Workshop on Noise Mapping and Environmental Acoustics*.
URL: http://www.csl.sony.fr/publications/item/?reference=defreville:06a

[118] Boris Defréville, Pierre Roy, Christophe Rosin, and Francois Pachet. "Automatic Recognition of Urban Sound Sources". In: *Presented at the 120th Audio Engineering Society (AES) Convention, May 20-23, 2006, Paris, France*.
URL: http://www.aes.org/e-lib/browse.cfm?elib=13631

[119] Boris Defréville et al. "Objective Representation of Urban Soundscape: Application to a Parisian Neighbourhood". In: *Proceedings of the 36th International Congress and Exposition on Noise Control Engineering (Inter-Noise 2007; August 28-31, 2007, Istambul, Turkey)*.
URL: http://www.lasa.fr/Actualites/internoise_2007.pdf

[120] Deredactie.be. *"Kleine handelaar uit de markt geprijsd"*. News report; In Dutch. VRT Nieuws. Apr. 23, 2009.
URL: http://deredactie.be/cm/vrtnieuws/archief/2.1222/economie/1.512348

[121] Deredactie.be. *Bonte stoet om Brugse crèche te steunen*. News report; In Dutch. VRT Nieuws. July 31, 2011.
URL: http://deredactie.be/cm/vrtnieuws/regio/westvlaanderen/1.1078807

[122] Anind K. Dey, Gregory D. Abowd, and Daniel Salber. "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications". In: *Human-Computer Interaction* 16.2 (Dec. 2001), pp. 97–166.
ISSN: 0737-0024
DOI: 10.1207/S15327051HCI16234_02

[123] Anind K. Dey, Daniel Salber, and Gregory D. Abowd. "A Context-based Infrastructure for Smart Environments". In: *Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments (MANSE 1999; Dublin, Ireland, December 13-14, 1999)*, pp. 114–128.
URL: http://www.cc.gatech.edu/fce/contexttoolkit/pubs/MANSE99-abstract.html

[124] Ellie D'Hondt and Luc Steels. *Community memories for sustainable urban living*. *BrusSense* project proposal. Granted as a "Prospective Research in Brussels" post-doctoral project funded by IWOIB/Innoviris. Vrije Universiteit Brussel, July 2009.
URL: http://www.brussense.be

[125] Ellie D'Hondt and Matthias Stevens. "Community memories for sustainable urban living". Poster abstract. In: *INNOV 2010 – Proceedings of the 1st International Multi-Conference on Innovative Developments in ICT (Athens, Greece, July 29-31, 2010)*. Ed. by Boris Shishkov, George A. Tsihrintzis, and Maria Virvou, pp. 77–80. Poster displayed at ICGREEN 2010 – International Conference on Green Computing (part of INNOV 2010).
ISBN: 978-989-8425-15-7
URL: http://soft.vub.ac.be/Publications/2010/vub-tr-soft-10-16.pdf

[126] Ellie D'Hondt and Matthias Stevens. "Participatory noise mapping". Demo abstract. In: *Demo Proceedings of the 9th International Conference on Pervasive Computing (Pervasive 2011), San Francisco, CA, USA, June 12-15, 2011*. Ed. by Rafael 'Tico' Ballagas and Daniela K. Rosner, pp. 33–36.
URL: http://soft.vub.ac.be/Publications/2011/vub-soft-tr-11-03.pdf

[127] Ellie D'Hondt, Matthias Stevens, and An Jacobs. "Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring". Preprint. Submitted to *Pervasive and Mobile Computing* (ISSN: 1574-1192). Conditionally accepted. Dec. 2011.
URL: http://www.noisetube.net/publications/partnoisemaps.pdf

[128] Thomas Dietz, Elinor Ostrom, and Paul C. Stern. "The Struggle to Govern the Commons". In: *Science* 302.5652 (Dec. 2003), pp. 1907–1912.
ISSN: 0036-8075
DOI: 10.1126/science.1091015

[129] Edsger Wybe Dijkstra. *A Discipline of Programming*. Prentice Hall, Oct. 1976. 217 pp.
ISBN: 978-0132158718

[130] Yong Ding, Friedemann Laue, Hedda R. Schmidtke, and Michael Beigl. "Sensing Spaces: Light-Weight Monitoring of Industrial Facilities". In: *Workshop on Behaviour Monitoring and Interpretation (BMI'10) held at the 33rd Annual German Conference on Artificial Intelligence (KI 2010; September 21, 2010, Karlsruhe, Germany)*.
URL: http://www.teco.edu/~ding

[131] *DIY Citizenship Conference & Hack Space – Critical Making & Social Media (November 11-14, 2010, Toronto, Canada)*.
URL: http://diycitizenship.com

[132] DIYSE project partners. *Do-it-Yourself Smart Experiences project (DiYSE)*. ITEA2 project.
URL: http://dyse.org

[133] Anhai Doan, Raghu Ramakrishnan, and Alon Y. Halevy. "Crowdsourcing Systems on the World-Wide Web". In: *Communications of the ACM* 54.4 (Apr. 2011), pp. 86–96.
ISSN: 0001-0782
DOI: 10.1145/1924421.1924442

[134] David Dodman. "Urban Form, Greenhouse Gas Emissions and Climate Vulnerability". In: *Population Dynamics and Climate Change*. Ed. by José Miguel Guzmán, George Martine, Gordon McGranahan, Daniel Schensul, and Cecilia Tacoli. United Nations Population Fund (UNFPA), International Institute for Environment, and Development (IIED), 2009. Chap. 4, pp. 64–79.
ISBN: 978-0-89714-919-8
URL: http://www.unfpa.org/public/home/publications/pid/4500

[135] George Drosatos, Pavlos S. Efraimidis, Ioannis N. Athanasiadis, Ellie D'Hondt, and Matthias Stevens. "A privacy-preserving cloud computing system for creating participatory noise maps". In: *COMPSAC 2012 – Trustworthy Software Systems for the Digital Society. IEEE Signature Conference on Computers, Software & Applications (Izmir, Turkey, July 16-20, 2012)*. Accepted for publication.

[136] Dani'ele Dubois, Catherine Guastavino, and Manon Raimbault. "A Cognitive Approach to Urban Soundscapes: Using Verbal Data to Access Everyday Life Auditory Categories". In: *Acta Acustica United with Acustica* 92.6 (Nov. 2006), pp. 865–874.
URL: http://www.ingentaconnect.com/content/dav/aaua/2006/00000092/00000006/art00003

[137] Christine E. Dunn. "Participatory GIS — a people's GIS?" In: *Progress in Human Geography* 31.5 (Oct. 2007), pp. 616–637.
ISSN: 0309-1325
DOI: 10.1177/0309132507081493

[138] Prabal Dutta et al. "Common Sense: Participatory Urban Sensing Using a Network of Handheld Air Quality Monitors". Demo Abstract. In: *SenSys '09: Proceedings of the 7th ACM Conference on Embedded Network Sensor Systems (Berkeley, CA, USA, November 4-6, 2009)*. New York, NY, USA: ACM, pp. 349–350.
ISBN: 978-1-60558-519-2
DOI: 10.1145/1644038.1644095

[139] Nathan Eagle, Alex (Sandy) Pentland, and David Lazer. "Inferring friendship network structure by using mobile phone data". In: *Proceedings of the National Academy of Sciences (PNAS)* 106.36 (Sept. 8, 2009), pp. 15274–15278.
ISSN: 0027-8424
DOI: 10.1073/pnas.0900282106

[140] Nathan Eagle and Alex (Sandy) Pentland. "Reality mining: sensing complex social systems". In: *Personal and Ubiquitous Computing* 10.4 (May 2006), pp. 255–268.
ISSN: 1617-4909
DOI: 10.1007/s00779-005-0046-3

[141] Eclipse Foundation. *Eclipse integrated development environment (IDE)*.
URL: http://www.eclipse.org

[142] "Mapping a better world". In: *The Economist* 391.8634 (June 6, 2009). Technology Quarterly, pp. 8–9.
ISSN: 0013-0613
URL: http://www.economist.com/node/13725877

[143] "Sensors and sensitivity". In: *The Economist* 391.8634 (June 6, 2009). Technology Quarterly, pp. 21–22.
ISSN: 0013-0613
URL: http://www.economist.com/node/13725679

[144] "Beyond the PC". Special Report on Personal Technology. In: *The Economist* 401.8754 (Oct. 8, 2011). After p. 58.
ISSN: 0013-0613
URL: http://www.economist.com/node/21531109

[145] "Internet blackouts – Reaching for the kill switch". International. In: *The Economist* 398.8720 (Feb. 12, 2011), pp. 67–68.
ISSN: 0013-0613
URL: http://www.economist.com/node/18112043

[146] "The Republicans – A dangerous game". Briefing. In: *The Economist* 401.8758 (Nov. 5, 2011), pp. 33–36.
ISSN: 0013-0613
URL: http://www.economist.com/node/21536603

[147] Shane B. Eisenman et al. *BikeNet and BikeView*. Dartmouth College.
URL: http://bikenet.cs.dartmouth.edu

[148] Shane B. Eisenman et al. "MetroSense Project: People-Centric Sensing at Scale". In: *WSW'06: Workshop on World-Sensor-Web (October 31, 2006), held at ACM SenSys '06 (October 31-November 3, 2006, Boulder, CO, USA)*.
URL: http://www.sensorplanet.org/wsw2006/11_Eisenman_WSW06_final.pdf

[149] Shane B. Eisenman et al. "The BikeNet Mobile Sensing System for Cyclist Experience Mapping". In: *SenSys '07 – Proceedings of the 5th international conference on Embedded networked sensor systems (Sydney, Australia, November 4-9, 2007)*. New York, NY, USA: ACM, pp. 87–101.
ISBN: 978-1-59593-763-6
DOI: 10.1145/1322263.1322273

[150] Shane B. Eisenman et al. "BikeNet: A mobile sensing system for cyclist experience mapping". In: *ACM Transactions on Sensor Networks (TOSN)* 6.1 (Jan. 2010), 6:1–6:39.
ISSN: 1550-4859
DOI: 10.1145/1653760.1653766

[151] Claire Ellul, Louise Francis, and Mordechai (Muki) Haklay. "A Flexible Database-Centric Platform for Citizen Science Data Capture". In: *Computing for Citizen Science Workshop (8 December 2011), held at IEEE eScience 2011, tyhe 7th IEEE e-Science conference (5-8 December 2011, Stockholm, Sweden)*.
URL: http://itee.uq.edu.au/~eresearch/workshops/compcitsci2011

[152] Environmental Systems Research Institute, Inc. *ESRI Shapefile Technical Description*. ESRI White Paper. July 1998. 34 pp.
URL: http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

[153] Deborah Estrin. "Participatory Sensing: Applications and Architecture". In: *IEEE Internet Computing* 14.1 (Jan. 2010). Part of "Internet Predictions" feature on the future of the Internet, pp. 12–14.
ISSN: 1089-7801
DOI: 10.1109/MIC.2010.12

[154] Commission of the European Communities. *Future Noise Policy*. Green paper COM(96) 540 final. Nov. 4, 1996.
ISBN: 92-78-10730-1
ISSN: 0254-1475
URL: http://ec.europa.eu/environment/noise/greenpap.htm

[155] European Commission. *Report from the Commission to the European Parliament and the Council on the implementation of the Environmental Noise Directive in accordance with Article 11 of Directive 2002/49/EC*. Report COM(2011) 321 final. June 1, 2011. 13 pp.
URL: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:52011DC0321:EN:NOT

[156] European Commission. *Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee of the Regions – Rio+20: towards the green economy and better governance*. Communication COM(2011) 363 final. June 20, 2011. 13 pp.
URL: http://ec.europa.eu/environment/international_issues/rio20_en.htm

[157] European Commission. "Commission Recommendation of 6 August 2003 concerning the guidelines on the revised interim computation methods for industrial noise, aircraft noise, road traffic noise and railway noise, and related emission data". In: *Official Journal of the European Union* L 212.46 (Aug. 22, 2003), pp. 49–64.
ISSN: 0378-6978
URL: http://ec.europa.eu/environment/noise/mapping.htm

[158] European Commission. *Attitudes on issues related to EU Energy Policy*. Analytical report. Flash Eurobarometer 206a. Fieldwork February 2007. Apr. 2007. 60 pp.
URL: http://ec.europa.eu/public_opinion/archives/flash_arch_207_192_en.htm

[159] Working Group – Assessment of Exposure to Noise (WG-AEN). *Good Practice Guide for Strategic Noise Mapping and the Production of Associated Data on Noise Exposure (Version 2)*. Position Paper. European Commission, Aug. 2007.
URL: http://ec.europa.eu/environment/noise/mapping.htm

[160] Scientific Committee on Emerging and Newly Identified Health Risks. *Potential health risks of exposure to noise from personal music players and mobile phones including a music playing function*. SCENIHR opinion. European Commission, Directorate-General for Health & Consumers, Sept. 23, 2008. 81 pp.
URL: http://ec.europa.eu/health/opinions/en/hearing-loss-personal-music-player-mp3

[161] European Commission. *Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee and the Committee of the Regions – Internet of Things: An action plan for Europe*. Communication COM(2009) 278 final. June 18, 2009. 11 pp.
URL: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:52009DC0278:EN:NOT

[162] European Commission. *Europeans' attitudes towards climate change*. Report. Special Eurobarometer 322. EB72.1. Fieldwork August–September 2009. Nov. 2009. 134 pp.
URL: http://ec.europa.eu/public_opinion/archives/eb_special_339_320_en.htm

[163] European Commission. *Attitudes of European citizens towards the environment*. Report. Special Eurobarometer 365. EB75.2. Fieldwork April–May 2011. June 2011. 178 pp.
URL: http://ec.europa.eu/public_opinion/archives/eb_special_379_360_en.htm

[164] European Commission. *Climate change*. Report. Special Eurobarometer 372. EB75.4. Fieldwork July 2011. Oct. 2011. 85 pp.
URL: http://ec.europa.eu/public_opinion/archives/eb_special_379_360_en.htm

[165] European Commission and European Parliament. *Europeans' attitudes towards climate change*. Report. Special Eurobarometer 300. EB69.2. Fieldwork March–May 2008. Sept. 2008. 145 pp.
URL: http://ec.europa.eu/public_opinion/archives/eb_special_320_300_en.htm

[166] European Commission and European Parliament. *Europeans' attitudes towards climate change*. Report. Special Eurobarometer 313. EB71.1. Fieldwork January–February 2009. July 2009. 67 pp.
URL: http://ec.europa.eu/public_opinion/archives/eb_special_320_300_en.htm

[167] European Commission – Environment DG. *Noise – Reporting on the implementation of the Directive*. 2012.
URL: http://ec.europa.eu/environment/noise/reporting.htm

[168] Council of the European Union. *Brussels European Council 8/9 March 2007 – Presidency Conclusions*. 7224/1/07 Rev 1. May 2, 2007. 25 pp.
URL: http://register.consilium.europa.eu/servlet/driver?typ=Simple&cmsid=638&lang=EN&fc=REGAISEN&ff_COTE_DOCUMENT=7224/1/07&rc=1&nr=1&page=Detail

[169] European Environment Agency (EEA), Microsoft Corporation, et al. *Eye on Earth*.
URL: http://www.eyeonearth.org

[170] European Environment Agency and Microsoft. *Eye on Earth Watches Solution Overview*. Dec. 2011. 4 pp.
URL: http://microsoft.com/publicsector/ww/international-organizations/projects/Pages/eye-on-earth.aspx

[171] European Foundation for the Improvement of Living and Working Conditions (Eurofound). *EurLIFE – Data for indicator: Complain about noise*. Based on Eurobarometer data (1992–2002).
URL: http://eurofound.europa.eu/areas/qualityoflife/eurlife/index.php?template=3&radioindic=96&idDomain=8

[172] European Foundation for the Improvement of Living and Working Conditions (Eurofound). *European Quality of Life Survey 2007 – Mapping the results*.
URL: http://www.eurofound.europa.eu/surveys/smt/eqls/results.htm

[173] European Parliament and Council. "Directive 2002/49/EC of 25 June 2002 relating to the assessment and management of environmental noise". In: *Official Journal of the European Communities* L 189.45 (July 18, 2002), pp. 12–26.
ISSN: 0378-6978
URL: http://ec.europa.eu/environment/noise/directive.htm

[174] European Parliament and Council. "Directive 2003/10/EC of 6 February 2003 on the minimum health and safety requirements regarding the exposure of workers to the risks arising from physical agents (noise)". In: *Official Journal of the European Union* L 42.46 (Feb. 15, 2003), pp. 38–44.
ISSN: 0378-6978
URL: http://osha.europa.eu/legislation/directives/exposure-to-physical-hazards/osh-directives/82

[175] EveryAware project partners. *EveryAware*. EU-funded research project (FP7).
URL: http://www.everyaware.eu

[176] Faber Acoustical, LLC. *dB*. iPhone app.
URL: http://www.faberacoustical.com/products/iphone/db

[177] Faber Acoustical, LLC. *SoundMeter*. iPhone app.
URL: http://www.faberacoustical.com/products/iphone/soundmeter

[178] Hugo Fastl and Eberhard Zwicker. *Psychoacoustics: Facts and Models*. 3rd ed. Springer Series in Information Sciences, vol. 22. Springer Berlin Heidelberg, 2007. 462 pp.
ISBN: 978-3-540-23159-2
DOI: 10.1007/978-3-540-68888-4

[179] Michela Ferron and Paolo Massa. "Collective memory building in Wikipedia: the case of North African uprisings". In: *WikiSym '11 – Proceedings of the 7th International Symposium on Wikis and Open Collaboration (Mountain View, CA, USA, October 3-5, 2011)*. New York, NY, USA: ACM, pp. 114–123.
ISBN: 978-1-4503-0909-7
DOI: 10.1145/2038558.2038578

[180] Luca Filipponi, Silvia Santini, and Andrea Vitaletti. "Data Collection in Wireless Sensor Networks for Noise Pollution Monitoring". In: *Distributed Computing in Sensor Systems: Proceedings of the 4th IEEE International Conference, DCOSS 2008 Santorini Island, Greece, June 11-14, 2008*. Lecture Notes in Computer Science, vol. 5067. Springer Berlin / Heidelberg, pp. 492–497.
ISBN: 978-3-540-69169-3
ISSN: 0302-9743
DOI: 10.1007/978-3-540-69170-9_35

[181] Harvey Fletcher and W. A. Munson. "Loudness, Its Definition, Measurement and Calculation". In: *Journal of the Acoustical Society of America* 5.2 (Oct. 1933), pp. 82–108.
ISSN: 0001-4966
DOI: 10.1121/1.1915637

[182] Mary Florentine, Arthur N. Popper, and Richard R. Fay, eds. *Loudness*. Springer Handbook of Auditory Research, vol. 37. Springer New York, 2011. 290 pp.
ISBN: 978-1-4419-6711-4
ISSN: 0947-2657
DOI: 10.1007/978-1-4419-6712-1

[183] Theodor Foerster et al. *OpenNoiseMap*. Institute for Geoinformatics, University of Münster. Oct. 2010.
URL: http://www.opennoisemap.org

[184] Theodor Foerster, Laura Díaz, and Arne Bröring. "Integrating Volunteered Human Sensor Data into Crowd-sourced Platforms: A Use Case on Noise Pollution Monitoring and OpenStreetMap". In: *Innovations in Sharing Environmental Observation and Information. Proceedings of the 25th EnviroInfo Conference "Environmental Informatics" (October 5-7, 2011, Ispra, Italy)*. Ed. by Werner Pillmann, Sven Schade, and Paul Smits. Shaker Publishing, pp. 505–510.
ISBN: 978-3-8440-0451-9
URL: http://ifgi.uni-muenster.de/~arneb/EnviroInfo2011_submitted.pdf

[185] Theodor Foerster et al. “Integrating Human Observations and Sensor Observations – the Example of a Noise Mapping Community”. In: *DE-2010 – Towards Digital Earth: Search, Discover and Share Geospatial Data 2010. Workshop held at Future Internet Symposium, Berlin, Germany, September 20, 2010*. CEUR Workshop Proceedings, vol. 640.
ISSN: 1613-0073
URL: http://ceur-ws.org/Vol-640/paper1.pdf

[186] B. J. Fogg. *Persuasive Technology: Using Computers to Change What We Think and Do*. Interactive Technologies. Morgan Kaufmann, 2002. 312 pp.
ISBN: 978-1-55860-643-2
URL: http://www.elsevierdirect.com/product.jsp?isbn=9781558606432

[187] Ian Bogost et al. *Mobile Persuasion: 20 Perspectives of the Future of Behavior Change*. Ed. by B.J. Fogg and Dean Eckles. Stanford Captology Media, 2007. 166 pp.
ISBN: 978-0-9795025-2-1
URL: http://mobilepersuasion.com

[188] Food and Agriculture Organization of the United Nations, Fisheries and Aquaculture Department. *The State of World Fisheries and Aquaculture – 2010*. Rome, Italy: FOA, 2010. 218 pp.
ISBN: 978-92-5-106675-1
ISSN: 1020-5489
URL: http://www.fao.org/docrep/013/i1820e/i1820e00.htm

[189] Marcus Foth, ed. *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*. Hershey, PA, USA / London, UK: Information Science Reference, IGI Global, 2009. 506 pp.
ISBN: 978-1-60566-152-0
DOI: 10.4018/978-1-60566-152-0
URL: http://www.vrolik.de/book

[190] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley Signature Series. Addison-Wesley Professional, Nov. 2002. 560 pp.
ISBN: 978-0321127426
URL: http://www.martinfowler.com/books.html#eaa

[191] Martin Fowler. *UML Distilled. A Brief Guide to the Standard Object Modeling Language*. 3rd ed. Object Technology Series. Boston, MA, USA: Addison-Wesley, Sept. 2003. 208 pp.
ISBN: 978-0321193681
URL: http://martinfowler.com/books.html#uml

[192] Martin Fowler, Kent Beck, John Brant, William Opdyke, and Don Roberts. *Refactoring: Improving the Design of Existing Code*. Ed. by Martin Fowler. Object Technology Series. Boston, MA, USA: Addison-Wesley, July 1999. 464 pp.
ISBN: 978-0201485677
URL: http://martinfowler.com/books.html#refactoring

[193] Louise Francis, Colleen Whitaker, and Muki Haklay. "Noise Mapping Helps Citizens Take Action". In: *GIS Professional* 23 (Aug. 2008), pp. 26–28.

[194] Free Software Foundation, Inc. *GNU General Public License (GPL), version 2*. 1989-1991.
URL: http://www.gnu.org/licenses/gpl-2.0.html

[195] Free Software Foundation, Inc. *GNU Lesser General Public License (LGPL), version 2.1*. Formerly the called the *GNU Library General Public License*. 1991-1999.
URL: http://www.gnu.org/licenses/lgpl-2.1.html

[196] Jon Froehlich et al. "UbiGreen: investigating a mobile tool for tracking and supporting green transportation habits". In: *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems (April 4-9, 2009, Boston, MA, USA)*. New York, NY, USA: ACM, pp. 1043–1052.
ISBN: 978-1-60558-246-7
DOI: 10.1145/1518701.1518861

[197] Kris Fudalewsk. *KJ Digital Signal Synchronizer (KJDSS) library*. Licensed under the GNU Lesser General Public License. 2010.
URL: http://sirk.sytes.net/software/libs/kjdss

[198] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series. Boston, MA, USA: Addison-Wesley, Oct. 1994. 416 pp. The authors, and the book itself, are often referred to as the *Gang of Four*, or *GoF*.
ISBN: 978-0201633610
URL: http://www.pearsoned.co.uk/bookshop/detail.asp?item=171742

[199] G. David Garson. *Computer Technology and Social Issues*. Hershey, PA, USA: Idea Group Publishing, Jan. 1995. 456 pp.
ISBN: 978-1878289285
DOI: 10.4018/978-1-87828-928-5

[200] Gartner, Inc. *Gartner Says Worldwide Mobile Phone Sales to End Users Grew 8 Per Cent in Fourth Quarter 2009; Market Remained Flat in 2009*. Press release. Feb. 23, 2010.
URL: http://www.gartner.com/it/page.jsp?id=1306513

[201] Gartner, Inc. *Gartner Says Worldwide Mobile Device Sales to End Users Reached 1.6 Billion Units in 2010; Smartphone Sales Grew 72 Percent in 2010*. Press release. Feb. 9, 2011.
URL: http://www.gartner.com/it/page.jsp?id=1543014

[202] Gartner, Inc. *Gartner Says 428 Million Mobile Communication Devices Sold Worldwide in First Quarter 2011, a 19 Percent Increase Year-on-Year*. Press release. May 19, 2011.
URL: http://www.gartner.com/it/page.jsp?id=1689814

[203] Gartner, Inc. *Gartner Says Sales of Mobile Devices in Second Quarter of 2011 Grew 16.5 Percent Year-on-Year; Smartphone Sales Grew 74 Percent*. Press release. Aug. 11, 2011.
URL: http://www.gartner.com/it/page.jsp?id=1764714

[204] Gartner, Inc. *Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent*. Press release. Nov. 15, 2011.
URL: http://www.gartner.com/it/page.jsp?id=1848514

[205] Gartner, Inc. *Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth*. Press release. Feb. 15, 2012.
URL: http://www.gartner.com/it/page.jsp?id=1924314

[206] Elisa Giaccardi et al. "Acoustic Cartographies: Supporting Experiential Learning and Shared Understanding of the Natural Heritage Through Collaborative Mapping". In: *Proceedings of the 19th International Congress on Acoustics (Madrid, Spain, 2-7 September 2007)*.
URL: http://www.thesilence.org/documents/SOL_AcousticCartographies.pdf

[207] Jonathan Gifford. *EU to launch tender for open data portal*. Deutsche Welle. Apr. 20, 2011.
URL: http://www.dw-world.de/dw/article/0,,15007160,00.html

[208] Jim Giles. "Internet encyclopaedias go head to head". Special report. In: *Nature* 438.7070 (Dec. 2005), pp. 900–901.
ISSN: 0028-0836
DOI: 10.1038/438900a

[209] Malcolm Gladwell. *Does Egypt Need Twitter?* The New Yorker. Feb. 2, 2011.
URL: http://www.newyorker.com/online/blogs/newsdesk/2011/02/does-egypt-need-twitter.html

[210] Lisa Goines and Louis Hagler. "Noise Pollution: A Modern Plague". Review Article. In: *Southern Medical Journal* 100.3 (Mar. 2007), pp. 287–294.
ISSN: 0038-4348
DOI: 10.1097/SMJ.0b013e3180318be5

[211] Scott A. Golder and Bernardo A. Huberman. "Usage patterns of collaborative tagging systems". In: *Journal of Information Science* 32.2 (Apr. 2006), pp. 198–208.
ISSN: 0165-5515
DOI: 10.1177/0165551506062337

[212] Marta C. Gonzalez, Cesar A. Hidalgo, and Albert-Laszlo Barabasi. "Understanding individual human mobility patterns". Letter. In: *Nature* 453.7196 (June 5, 2008), pp. 779–782.
ISSN: 0028-0836
DOI: 10.1038/nature06958

[213] Michael F. Goodchild. "Citizens as sensors: the world of volunteered geography". In: *GeoJournal* 69.4 (Aug. 2007), pp. 211–221.
ISSN: 0343-2521
DOI: 10.1007/s10708-007-9111-y

[214] Google, Inc. *App Inventor for Android*.
URL: http://www.appinventorbeta.com/about

[215] Google, Inc. *Google Earth*.
URL: http://earth.google.com

[216] Google, Inc. *Google Maps*.
URL: http://maps.google.com

[217] Google, Inc. *Google Maps API*.
URL: http://code.google.com/apis/maps

[218] Google, Inc. *Maps External Library*. External library for Android.
URL: http://code.google.com/android/add-ons/google-apis/maps-overview.html

[219] Google, Inc. *Play*. Formerly called *Android Market*.
URL: http://play.google.com

[220] Google, Inc. *Google to Acquire Motorola Mobility*. Message released to investors. Aug. 15, 2011.
URL: http://investor.google.com/releases/2011/0815.html

[221] Google, Inc. and independent contributors. *Scripting Layer for Android (SL4A)*.
URL: http://code.google.com/p/android-scripting

[222] Google, Inc. and Open Handset Alliance. *Android Development Tools (ADT) plugin for Eclipse*.
URL: http://developer.android.com/sdk/eclipse-adt.html

[223] Google, Inc. and Open Handset Alliance. *Android Dev Guide: Android API Levels*.
URL: http://developer.android.com/guide/appendix/api-levels.html

[224] Google, Inc. and Open Handset Alliance. *Android Dev Guide: What is Android?*
URL: http://developer.android.com/guide/basics/what-is-android.html

[225] Google, Inc. and Open Handset Alliance. *Android Native Development Kit (NDK)*.
URL: http://developer.android.com/sdk/ndk

[226] Google, Inc. and Open Handset Alliance. *Android open source project*.
URL: http://www.android.com

[227] Google, Inc. and Open Handset Alliance. *Android Software Development Kit (SDK)*.
URL: http://developer.android.com/sdk

[228] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *The Java™ Language Specification*. 2nd ed. The Java™ Series. Addison Wesley, June 2000. 544 pp.
ISBN: 978-0201310085
URL: http://java.sun.com/docs/books/jls

[229] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *The Java™ Language Specification*. 3rd ed. The Java™ Series. Addison Wesley, June 2005. 688 pp.
ISBN: 978-0-321-24678-3
URL: http://java.sun.com/docs/books/jls

[230] Catherine Guastavino. "The ideal urban soundscape: Investigating the sound quality of French cities". In: *Acta Acustica United with Acustica* 92.6 (Nov. 2006), pp. 945–951.

[231] Catherine Guastavino. "Categorization of environmental sounds". In: *Canadian Journal of Experimental Psychology* 61.1 (Mar. 2007), pp. 54–63.
ISSN: 1196-1961
DOI: 10.1037/cjep2007006

[232] Davis Guggenheim and Al Gore. *An Inconvenient Truth*. Documentary film directed Davis Guggenheim, written by and starring Al Gore. 2006.
URL: http://www.climatecrisis.net

[233] Erik Gunderson, Jacqueline Moline, and Peter Catalano. "Risks of Developing Noise-Induced Hearing Loss in Employees of Urban Music Clubs". In: *American Journal of Industrial Medicine* 31.1 (Jan. 1997), pp. 75–79.
ISSN: 0271-3586
DOI: 10.1002/(SICI)1097-0274(199701)31:1<75::AID-AJIM11>3.0.CO;2-4

[234] Vehbi C. Gungor and Gerhard P. Hancke. "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches". In: *IEEE Transactions on Industrial Electronics* 56.10 (Oct. 2009), pp. 4258–4265.
ISSN: 0278-0046
DOI: 10.1109/TIE.2009.2015754

[235] HabitatMap and Lunar Logic Polska. *AirCasting*.
URL: http://aircasting.org

[236] Mordechai Haklay. "How good is volunteered geographical information? A comparative study of OpenStreetMap and Ordnance Survey datasets". In: *Environment and Planning B: Planning and Design* 37.4 (2010), pp. 682–703.
DOI: 10.1068/b35097

[237] Mordechai (Muki) Haklay. "Geographical Citizen Science – Clash of cultures and new opportunities". Position paper. In: *Workshop on the role of volunteered geographic information in advancing science (September 14, 2010), held in conjunction with GIScience 2010 (14-17 September, 2010, Zurich, Switzerland)*.
URL: http://www.ornl.gov/sci/gist/workshops/papers/Haklay.pdf

[238] Mordechai (Muki) Haklay and Claire Ellul. "Completeness in volunteered geographical information – the evolution of OpenStreetMap coverage in England (2008-2009)". In: *Journal of Spatial Information Science* (2011). In revision.
ISSN: 1948-660X
URL: http://www.josis.org/index.php/josis/article/view/35/35

[239] Mordechai (Muki) Haklay and Patrick Weber. "OpenStreetMap: User-Generated Street Maps". In: *IEEE Pervasive Computing* 7.4 (Oct. 2008), pp. 12–18.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2008.80

[240] Muki Haklay, Alex Singleton, and Chris Parker. "Web Mapping 2.0: The Neogeography of the GeoWeb". In: *Geography Compass* 2.6 (Nov. 2008), pp. 2011–2039.
ISSN: 1749-8198
DOI: 10.1111/j.1749-8198.2008.00167.x

[241] Wouter Halfwerk, Leonard J. M. Holleman, C(Kate). M. Lessells, and Hans Slabbekoorn. "Negative impact of traffic noise on avian reproductive success". In: *Journal of Applied Ecology* 48.1 (Feb. 2011), pp. 210–219.
ISSN: 1365-2664
DOI: 10.1111/j.1365-2664.2010.01914.x

[242] Peter Hanappe, Anthony Beurivé, Florence Laguzet, Luc Steels, Nicolas Bellouin, Olivier Boucher, Yasuhiro Hiro Yamazaki, Tolu Aina, and Myles R. Allen. "FAMOUS, faster: using parallel computing techniques to accelerate the FAMOUS/HadCM3 climate model with a focus on the radiative transfer algorithm". In: *Geoscientific Model Development* 4.3 (Sept. 2011), pp. 835–844.
ISSN: 1991-959X
DOI: 10.5194/gmd-4-835-2011

[243] Eric Hand. "Citizen science: People power". News Feature. In: *Nature* 466.7307 (Aug. 2010), pp. 685–687.
ISSN: 0028-0836
DOI: 10.1038/466685a

[244] Usman Haque. *Extended Environments Markup Language (EEML)*. Haque Design + Research Ltd. 2007–2008.
URL: http://www.eeml.org

[245] Garrett Hardin. "The Tragedy of the Commons". In: *Science* 162.3859 (Dec. 1968), pp. 1243–1248.
ISSN: 0036-8075
DOI: 10.1126/science.162.3859.1243

[246] Garrett Hardin. "Extensions of "The Tragedy of the Commons"". In: *Science* 280.5364 (May 1998), pp. 682–683.
ISSN: 0036-8075
DOI: 10.1126/science.280.5364.682

[247] Garrett Hardin. "Tragedy of the Commons". In: *The Concise Encyclopedia of Economics*. Ed. by David R. Henderson. Liberty Fund, 2008, pp. 497–499.
ISBN: 978-0-86597-666-5
URL: http://www.econlib.org/library/Enc/TragedyoftheCommons.html

[248] Anne-Wil Harzing. *Publish or Perish*. Software program that retrieves and analyzes academic citations.
URL: http://www.harzing.com/pop.htm

[249] David Heinemeier Hansson et al. *Active Record — Object-relation mapping put on rails*.
URL: http://ar.rubyonrails.org

[250] David Heinemeier Hansson, Rails Core Team, et al. *Ruby on Rails*.
URL: http://rubyonrails.org

[251] Tom Henderson and the Physics Classroom Project. *The Physics Classroom*. 1996-2011.
URL: http://www.physicsclassroom.com

[252] Jeffrey Hightower, Anthony LaMarca, and Ian E. Smith. "Practical Lessons from Place Lab". In: *IEEE Pervasive Computing* 5.3 (July 2006), pp. 32–39.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2006.55

[253] Robert Hirschfeld, Pascal Costanza, and Oscar Nierstrasz. "Context-oriented Programming". In: *Journal of Object Technology* 7.3 (Mar. 2008), pp. 125–151.
ISSN: 1660-1769
DOI: 10.5381/jot.2008.7.3.a4

[254] Richard Honicky, Eric A. Brewer, Eric Paulos, and Richard White. "N-SMARTS: Networked Suite of Mobile Atmospheric Real-time Sensors". In: *Proceedings of the second ACM SIGCOMM workshop on Networked systems for developing regions (NSDR '08)*. Held at ACM SIGCOMM 2008 Conference (Seattle, WA, USA, August 17-22, 2008). New York, NY, USA: ACM, pp. 25–30.
ISBN: 978-1-60558-180-4
DOI: 10.1145/1397705.1397713

[255] Richard Edward Honicky. *Automatic calibration of sensor-phones using gaussian processes*. Tech. rep. UCB/EECS-2007-34. EECS Department, University of California, Berkeley, Mar. 2007.
URL: http://www.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-34.html

[256] R.J. Honicky, Eric A. Brewer, John F. Canny, and Ronald C. Cohen. "Increasing the precision of mobile sensing systems through super-sampling". In: *UrbanSense08: Proceedings of International Workshop on Urban, Community, and Social Applications of Networked Sensing Systems (November 4, 2008; Raleigh, NC, USA)*. Held in conjunction with ACM SenSys 2008 (Raleigh, NC, USA, November 4 -7, 2008), pp. 31–35.
URL: http://sensorlab.cs.dartmouth.edu/urbansensing/papers/honicky_urbansense08.pdf

[257] Michael Hopkin. "Conservation: Mark of respect". News Feature. In: *Nature* 448.7152 (July 2007), pp. 402–403.
ISSN: 0028-0836
DOI: 10.1038/448402a

[258] Teerayut Horanont and Ryosuke Shibasaki. "An Implementation of Mobile Sensing for Large-Scale Urban Monitoring". In: *UrbanSense08: Proceedings of International Workshop on Urban, Community, and Social Applications of Networked Sensing Systems (November 4, 2008; Raleigh, NC, USA)*. Held in conjunction with ACM SenSys 2008 (Raleigh, NC, USA, November 4 -7, 2008), pp. 51–55.
URL: http://sensorlab.cs.dartmouth.edu/urbansensing/papers/horanont_urbansense08.pdf

[259] Jeff Howe. "The Rise of Crowdsourcing". In: *Wired* 14.06 (June 2006).
ISSN: 1059-1028
URL: http://www.wired.com/wired/archive/14.06/crowds.html

[260] HTC Corporation. *HTC Desire Z*. 2010.
URL: http://www.htc.com/europe/smartphones/htc-desire-z

[261] Kuan Lun Huang, Salil S. Kanhere, and Wen Hu. "Are You Contributing Trustworthy Data? The Case for a Reputation System in Participatory Sensing". In: *MSWIM '10 – Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems (October 17-21, 2010, Bodrum, Turkey)*. New York, NY, USA: ACM, pp. 14–22.
ISBN: 978-1-4503-0274-6
DOI: 10.1145/1868521.1868526

[262] Emmanuel C. Ifeachor and Barrie W. Jervis. *Digital Signal Processing: A Practical Approach*. 2nd ed. Pearson Education, Oct. 2001. 960 pp.
ISBN: 978-0201596199
URL: http://www.pearsoned.co.uk/bookshop/detail.asp?item=100000000003290

[263] Imperial College London. *EpiCollect. Mobile/Web Application for Smartphone data collection*.
URL: http://www.epicollect.net

[264] International Electrotechnical Commission. *Frequency weighting for the measurement of aircraft noise (D-weighting)*. IEC 60537:1976. Jan. 1976. Withdrawn.
URL: http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/24969

[265] International Electrotechnical Commission. *Sound level meters*. IEC 60651:2001. 1979-2001. 77 pp. Withdrawn, replaced by IEC 61672-1:2002 and IEC 61672-2:2003.
URL: http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/27971

[266] International Electrotechnical Commission. *Integrating-averaging sound level meters*. IEC 60804:2000. 1985-2000. 59 pp. Withdrawn, replaced by IEC 61672-1:2002 and IEC 61672-2:2003.
URL: http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/26439

[267] International Electrotechnical Commission. *Electroacoustics – Specifications for personal sound exposure meters*. IEC 61252:2002. 1993-2002. 59 pp. Adopted by the European Committee for Electrotechnical Standardization (CENELEC) as EN 61252:1995/A1:2001.
URL: http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/28582

[268] International Electrotechnical Commission. *Electroacoustics – Sound level meters – Part 1: Specifications*. IEC 61672-1:2002. May 2002. 35 pp. Adopted by the European Committee for Electrotechnical Standardization (CENELEC) as EN 61672-1:2003. Mentioned page numbers refer to the EN 61672-1:2003 document.
URL: http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/28763

[269] International Organization for Standardization. *Acoustics – Method for calculating loudness level*. ISO 532:1975. 1975. 18 pp.
URL: http://www.iso.org/iso/catalogue_detail?csnumber=4603

[270] International Organization for Standardization. *Acoustics – Determination of occupational noise exposure and estimation of noise-induced hearing impairment*. ISO 1999:1990. 1990. 17 pp.
URL: http://www.iso.org/iso/catalogue_detail?csnumber=6759

[271] International Organization for Standardization. *Acoustics – Normal equal-loudness-level contours*. ISO 226:2003. 2003. 18 pp.
URL: http://www.iso.org/iso/catalogue_detail?csnumber=34222

[272] International Organization for Standardization. *Acoustics – Description, measurement and assessment of environmental noise – Part 1: Basic quantities and assessment procedures*. ISO 1996-1:2003. 2003. 26 pp.
URL: http://www.iso.org/iso/catalogue_detail?csnumber=28633

[273] International Organization for Standardization. *Acoustics – Description, measurement and assessment of environmental noise – Part 2: Determination of environmental noise levels*. ISO 1996-2:2007. 2007. 40 pp.
URL: http://www.iso.org/iso/catalogue_detail?csnumber=41860

[274] International Telecommunication Union. *The Internet of Things*. ITU Internet Reports 2005. Nov. 2005. 200 pp.
ISBN: 92-61-11291-9
URL: http://www.itu.int/internetofthings

[275] International Telecommunication Union. *The World in 2011 – ICT Facts and Figures*. Oct. 2011. 8 pp.
URL: http://www.itu.int/ITU-D/ict/facts/2011

[276] Intergovernmental Panel on Climate Change. *Climate Change 2007: Synthesis Report*. Ed. by IPCC Core Writing Team, Rajendra K. Pachauri, and Andy Reisinger. IPCC Fourth Assessment Report (AR4). 2007.
URL: http://www.ipcc.ch/publications_and_data/publications_ipcc_fourth_assessment_report_synthesis_report.htm

[277] i-SCOPE Consortium. *i-SCOPE Project: interoperable Smart City services through an Open Platform for urban Ecosystems*. CIP / ICT PSP project funded by the European Commission. 2012–2014.
URL: http://www.iscopeproject.net

[278] E. J. M. Jansen, H. W. Helleman, W. A. Dreschler, and J. A. P. M. de Laat. “Noise induced hearing loss and other hearing complaints among musicians of symphony orchestras”. In: *International Archives of Occupational and Environmental Health* 82.2 (2009), pp. 153–164.
ISSN: 0340-0131
DOI: 10.1007/s00420-008-0317-1

[279] Java Community Process. *JSR-30: Connected Limited Device Configuration (CLDC) v1.0*. Specification development led by Sun Microsystems, Inc. 2000.
URL: http://jcp.org/en/jsr/summary?id=30

[280] Java Community Process. *JSR-37: Mobile Information Device Profile (MIDP) v1.0a*. Specification developed by the Mobile Information Profile Expert Group (MIDPEG), led by Sun Microsystems, Inc. and Motorola, Inc. 2000.
URL: http://jcp.org/en/jsr/summary?id=37

[281] Java Community Process. *JSR-75: PDA Optional Packages*. Specification development led by PalmSource, Inc. Consists of the *File Connection Optional Package* (FCOP) and the *PIM Optional Package* (PIMOP). 2004.
URL: http://jcp.org/en/jsr/summary?id=75

[282] Java Community Process. *JSR-118: Mobile Information Device Profile (MIDP) v2.0 & v2.1*. Specification developed by the JSR-118 Expert Group, led by Sun Microsystems, Inc. and Motorola, Inc. 2002-2006.
URL: http://jcp.org/en/jsr/summary?id=118

[283] Java Community Process. *JSR-135: Mobile Media API (MMAPI)*. Specification developed by the MMAPI Expert Group, led by Nokia Corporation. 2002-2006.
URL: http://jcp.org/en/jsr/summary?id=135

[284] Java Community Process. *JSR-139: Connected Limited Device Configuration (CLDC) v1.1 & v1.1.1*. Specification development led by Sun Microsystems, Inc. 2003-2007.
URL: http://jcp.org/en/jsr/summary?id=139

[285] Java Community Process. *JSR-179: Location API*. Specification developed by the JSR-179 Expert Group, led by Nokia Corporation. 2003-2006.
URL: http://jcp.org/en/jsr/summary?id=179

[286] Java Community Process. *JSR-185: Java Technology for the Wireless Industry (JTWI)*. Specification development led by Sun Microsystems, Inc. 2003.
URL: http://jcp.org/en/jsr/summary?id=185

[287] Java Community Process. *JSR-248: Mobile Service Architecture (MSA)*. Specification development led by Nokia Corporation and Vodafone Group Services, Ltd. 2006-2008.
URL: http://jcp.org/en/jsr/summary?id=248

[288] Java Community Process. *JSR-271: Mobile Information Device Profile (MIDP) v3.0*. Specification development led by Motorola, Inc. 2009.
URL: http://jcp.org/en/jsr/summary?id=271

[289] August Joki, Jeffrey A. Burke, and Deborah Estrin. *Campaignr: A Framework for Participatory Data Collection on Mobile Phones*. CENS Paper 770. UC Los Angeles, Center for Embedded Network Sensing, Oct. 2007.
URL: http://escholarship.org/uc/item/8v01m8wj

[290] Kenneth Kaliski, Eddie Duncan, and James Cowan. "Community and Regional Noise Mapping in the United States". In: *Sound and Vibration* (Sept. 2007), pp. 14–17.
ISSN: 1541-0161
URL: http://www.sandv.com/sep07.shtml

[291] N. A. Kane, M. C. Simmons, D. John, D. L. Thompson, and D. R. Basset. "Validity of the Nike+Device During Walking and Running". In: *International Journal of Sports Medicine* 31.2 (Feb. 2010), pp. 101–105.
ISSN: 0172-4622
DOI: 10.1055/s-0029-1242810

[292] Eiman Kanjo. "NoiseSPY: A Real-Time Mobile Phone Platform for Urban Noise Monitoring and Mapping". In: *Mobile Networks and Applications* 15.4 (Aug. 2010), pp. 562–574.
ISSN: 1383-469X
DOI: 10.1007/s11036-009-0217-y

[293] Eiman Kanjo, Jean Bacon, David Roberts, and Peter Landshoff. "MobSens: Making Smart Phones Smarter". In: *IEEE Pervasive Computing* 8.4 (Oct. 2009), pp. 50–57.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2009.79

[294] Eiman Kanjo et al. "MobGeoSen: facilitating personal geosensor data collection and visualization using mobile phones". In: *Personal and Ubiquitous Computing* 12.8 (Nov. 2008), pp. 599–607.
ISSN: 1617-4909
DOI: 10.1007/s00779-007-0180-1

[295] Apu Kapadia, David Kotz, and Nikos Triandopoulos. "Opportunistic Sensing: Security Challenges for the New Paradigm". In: *COMSNETS 2009: First International Conference on COMmunication Systems and NETworkS (5-10 January, 2009, Bangalore, India)*.
ISBN: 978-1-4244-2912-7
DOI: 10.1109/COMSNETS.2009.4808850

[296] Apu Kapadia, Nikos Triandopoulos, Cory Cornelius, Daniel Peebles, and David Kotz. "AnonySense: Opportunistic and Privacy-Preserving Context Collection". In: *Pervasive Computing. 6th International Conference, Pervasive 2008 Sydney, Australia, May 19-22, 2008. Proceedings.* Ed. by Jadwiga Indulska, Donald Patterson, Tom Rodden, and Max Ott. Lecture Notes in Computer Science, vol. 5013. Springer Berlin / Heidelberg, pp. 280–297.
ISBN: 978-3-540-79575-9
ISSN: 0302-9743
DOI: 10.1007/978-3-540-79576-6_17

[297] Garret Keizer. *The Unwanted Sound of Everything We Want. A Book About Noise*. PublicAffairs, May 2010. 400 pp.
ISBN: 978-1586485528
URL: http://www.publicaffairsbooks.com/publicaffairsbooks-cgi-bin/display?book=9781586485528

[298] Keyhole, Inc. and Google, Inc. *KML*. Formerly *Keyhole Markup Language*. Standardised by the Open Geospatial Consortium (OGC).
URL: http://code.google.com/apis/kml

[299] Habibul Haque Khondker. "Role of the New Media in the Arab Spring". In: *Globalizations* 8.5 (Oct. 2011), pp. 675–679.
ISSN: 1474-7731
DOI: 10.1080/14747731.2011.621287

[300] Tim Kindberg, Matthew Chalmers, and Eric Paulos. "Urban Computing". Guest Editors' Introduction. In: *IEEE Pervasive Computing* 6.3 (July 2007), pp. 18–20.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2007.57

[301] Rachel King. *10 Web 2.0 Ideas that Failed*. Fast Company. July 11, 2008.
URL: http://www.fastcompany.com/articles/2008/07/10-web-ideas-that-failed.html

[302] Stephen F. King and Paul Brown. "Fix My Street or Else: Using the Internet to Voice Local Public Service Concerns". In: *ICEGOV '07 – Proceedings of the 1st international conference on Theory and practice of electronic governance (Macao SAR, China, December 10-13, 2007)*. New York, NY, USA: ACM, pp. 72–80.
ISBN: 978-1-59593-822-0
DOI: 10.1145/1328057.1328076

[303] Peter Kinnaird. "An interview with Dr. Beth Simone Noveck". In: *XRDS* 18.2 (Winter 2011), pp. 16–17.
ISSN: 1528-4972
DOI: 10.1145/2043236.2043245

[304] Andreas Krause, Eric Horvitz, Aman Kansal, and Feng Zhao. "Toward Community Sensing". In: *IPSN '08: International Conference on Information Processing in Sensor Networks (22-24 April, 2008, St. Louis, MO, USA)*, pp. 481 –492.
ISBN: 978-0-7695-3157-1
DOI: 10.1109/IPSN.2008.37

[305] Stacey Kuznetsov and Eric Paulos. "Participatory Sensing in Public Spaces: Activating Urban Surfaces with Sensor Probes". In: *DIS '10: Proceedings of the 8th ACM Conference on Designing Interactive Systems (Aarhus, Denmark, August 16-20, 2010)*. New York, NY, USA: ACM, pp. 21–30.
ISBN: 978-1-4503-0103-9
DOI: 10.1145/1858171.1858175

[306] Stacey Kuznetsov and Eric Paulos. "Rise of the Expert Amateur: DIY Projects, Communities, and Cultures". In: *NordiCHI '10: Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries (October 16-20, 2010, Reykjavik, Iceland)*. New York, NY, USA: ACM, pp. 295–304.
ISBN: 978-1-60558-934-3
DOI: 10.1145/1868914.1868950

[307] Stacey Kuznetsov and Eric Paulos. "UpStream: Motivating Water Conservation with Low-Cost Water Flow Sensing and Persuasive Displays". In: *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems (April 10-15, 2010, Atlanta, GA, USA)*. New York, NY, USA: ACM, pp. 1851–1860.
ISBN: 978-1-60558-929-9
DOI: 10.1145/1753326.1753604

[308] Roberta Kwok. "Personal technology: Phoning in data". News Feature. In: *Nature* 458.7241 (Apr. 2009), pp. 959–961.
ISSN: 0028-0836
DOI: 10.1038/458959a

[309] Nicholas D. Lane, Shane B. Eisenman, Mirco Musolesi, Emiliano Miluzzo, and Andrew T. Campbell. "Urban Sensing Systems: Opportunistic or Participatory?" In: *HotMobile '08: Proceedings of the 9th Workshop on Mobile Computing Systems and Applications (February 25-26, 2008, Napa Valley, CA, USA)*. New York, NY, USA: ACM, pp. 11–16.
ISBN: 978-1-60558-118-7
DOI: 10.1145/1411759.1411763

[310] Nicholas D. Lane et al. "A Survey of Mobile Phone Sensing". In: *IEEE Communications Magazine* 48.9 (Sept. 2010), pp. 140–150.
ISSN: 0163-6804
DOI: 10.1109/MCOM.2010.5560598

[311] Catherine Lavandier et al. *CART-ASUR – Representations CARTographiques de la qualite des Ambiances Sonores URbaines : acceptabilite des cartes*. Project proposal (accepted). In French. CART-ASUR consortium, Mar. 2011.

[312] Catherine Lavandier and Boris Defréville. "The Contribution of Sound Source Characteristics in the Assessment of Urban Soundscapes". In: *Acta Acustica United with Acustica* 92.6 (Nov. 2006), pp. 912–921.
URL: http://www.ingentaconnect.com/content/dav/aaua/2006/00000092/00000006/art00009

[313] Jeroen Lavrijsen. *Langetermijnmetingen wegverkeersgeluid – update 2012. Analyse van de resultaten van meetpost NMT6 (Wetteren, E40), 1992-2011*. Vlaamse overheid, Dept. LNE, Afdeling Lucht, Hinder, Risicobeheer, Milieu & Gezondheid, Dienst Hinder & Risicobeheer, Mar. 2012. 68 pp.
URL: http://www.lne.be/themas/hinder-en-risicos/geluidshinder/beleid/geluidmeetnet/wegverkeerslawaai

[314] Lawrence Lessig et al. *Creative Commons*.
URL: http://creativecommons.org

[315] Lawrence Lessig. *The Future of Ideas: The Fate of the Commons in a Connected World*. Random House, Oct. 2001. 368 pp.
ISBN: 978-0375505782
URL: http://www.the-future-of-ideas.com

[316] Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello. "A Practical Approach to Recognizing Physical Activities". In: *Pervasive Computing. 4th International Conference, PERVASIVE 2006, Dublin, Ireland, May 7-10, 2006. Proceedings*. Ed. by Kenneth Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley. Lecture Notes in Computer Science=, vol. 3968. Springer Berlin / Heidelberg, pp. 1–16.
ISBN: 978-3-540-33894-9
ISSN: 0302-9743
DOI: 10.1007/11748625_1

[317] Joe Levi. *What Can You Do With a Barometer on a Smartphone?* pocketnow.com, Corey Online Media, LLC. Oct. 19, 2011.
URL: http://pocketnow.com/android/what-can-you-do-with-a-barometer-on-a-smartphone

[318] Jerome Lewis. "Enabling forest people to map their resources & monitor illegal logging in Cameroon". News. In: *Before Farming* 2007.2 (Oct. 2007), p. 7.
ISSN: 1476-4261
URL: http://www.waspjournals.com/journals/beforefarming/journal_20072/news/index.php

[319] Sheng Liang. *The Java™ Native Interface: Programmer's Guide and Specification*. The Java™ Series. Prentice Hall, June 1999. 320 pp.
ISBN: 978-0201325775
URL: http://java.sun.com/docs/books/jni

[320] Gaetano Licitra, ed. *Noise Mapping in the EU. Models and Procedures*. CRC Press, Aug. 2012.
ISBN: 978-0415585095
URL: http://www.crcpress.com/product/isbn/9780415585095

[321] Tim Lindholm and Frank Yellin. *The Java™ Virtual Machine Specification*. 2nd ed. The Java™ Series. Prentice Hall, Apr. 1999. 496 pp.
ISBN: 978-0201432947
URL: http://java.sun.com/docs/books/jvms

[322] LMS International. *SCADAS III*.
URL: http://www.lmsintl.com/test/scadasIII

[323] LMS International. *Test.Lab*.
URL: http://www.lmsintl.com/test-lab

[324] London 21. *'Citizen Science' takes off as Residents tackle Neighbourhood Noise*. Press release. 2008.
URL: http://www.london21.org/page/79/project/show/wwwlondon21orgpage35show1501

[325] London 21 and University College London. *Mapping for Change*.
URL: http://www.mappingforchange.org.uk

[326] Hong Lu, Nicholas D. Lane, Shane B. Eisenman, and Andrew T. Campbell. "Bubble-sensing: Binding sensing tasks to the physical world". In: *Pervasive and Mobile Computing* 6.1 (Feb. 2010), pp. 58–71.
ISSN: 1574-1192
DOI: 10.1016/j.pmcj.2009.10.005

[327] Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. "SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones". In: *MobiSys '09: Proceedings of the 7th international conference on Mobile systems, applications, and services (Kraków, Poland, June 22-25, 2009)*. New York, NY, USA: ACM, pp. 165–178.
ISBN: 978-1-60558-566-6
DOI: 10.1145/1555816.1555834

[328] Hong Lu et al. "The Jigsaw Continuous Sensing Engine for Mobile Phone Applications". In: *SenSys '10: Proceedings of the 8th ACM Conference on Embedded Network Sensor Systems (Zurich, Switzerland, November 3-5, 2010)*. New York, NY, USA: ACM, pp. 71–84.
ISBN: 978-1-4503-0344-6
DOI: 10.1145/1869983.1869992

[329] Kurt Luther, Scott Counts, Kristin B. Stecher, Aaron Hoff, and Paul Johns. "Pathfinder: An Online Collaboration Environment for Citizen Scientists". In: *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems (April 4-9, 2009, Boston, MA, USA)*. New York, NY, USA: ACM, pp. 239–248.
ISBN: 978-1-60558-246-7
DOI: 10.1145/1518701.1518741

[330] David J.C. MacKay. *Sustainable Energy - Without the Hot Air*. UIT Cambridge, Dec. 2008. 380 pp.
ISBN: 978-0954452933
URL: http://www.withouthotair.com

[331] Nicolas Maisonneuve, Matthias Stevens, Maria E. Niessen, Peter Hanappe, and Luc Steels. "Citizen Noise Pollution Monitoring". In: *dg.o '09 – Proceedings of the 10th Annual International Conference on Digital Government Research (Puebla, Mexico; May 17-21, 2009). Social Networks: Making Connections between Citizens, Data and Government*. Ed. by Soon Ae Chun, Rodrigo Sandoval, and Priscilla Regan. ACM International Conference Proceeding Series, vol. 390. Digital Government Society of North America, pp. 96–103.
ISBN: 978-1-60558-535-2
URL: http://portal.acm.org/citation.cfm?id=1556176.1556198

[332] Nicolas Maisonneuve, Matthias Stevens, Maria E. Niessen, and Luc Steels. "NoiseTube: Measuring and mapping noise pollution with mobile phones". In: *Information Technologies in Environmental Engineering. Proceedings of the 4th International ICSC Symposium (ITEE 2009; Thessaloniki, Greece, May 28-29, 2009)*. Ed. by Ioannis N. Athanasiadis, Pericles A. Mitkas, Andrea E. Rizzoli, and Jorge Marx Gómez. Environmental Science and Engineering. Springer Berlin Heidelberg, pp. 215–228.
ISBN: 978-3-540-88350-0
ISSN: 1863-5520
DOI: 10.1007/978-3-540-88351-7_16
URL: http://soft.vub.ac.be/Publications/2009/vub-prog-tr-09-08.pdf

[333] Nicolas Maisonneuve, Matthias Stevens, and Bartek Ochab. "NoiseTube: Citizen Noise Pollution Monitoring". Poster displayed at the Sony CSL Paris Open House 2009 (Paris, France, 8-9 October, 2009).
URL: http://soft.vub.ac.be/Publications/2009/vub-prog-tr-09-11.pdf

[334] Nicolas Maisonneuve, Matthias Stevens, and Bartek Ochab. "Participatory noise pollution monitoring using mobile phones". In: *Information Polity* 15.1-2 (Aug. 2010): *Special Issue on "Government 2.0: Making Connections between Citizens, Data and Government"*, pp. 51–71.
ISSN: 1570-1255
DOI: 10.3233/IP-2010-0200
URL: http://soft.vub.ac.be/Publications/2010/vub-tr-soft-10-14.pdf

[335] Nicolas Maisonneuve, Matthias Stevens, and Luc Steels. "Measure and map noise pollution with your mobile phone". Instructable. In: *DIY::HCI. A Showcase of Methods, Communities and Values for Reuse and Customization*. Ed. by Leah Buechley, Eric Paulos, Daniela Rosner, and Amanda Williams. Proceedings of the "DIY for CHI" workshop held at CHI '09, the 27th International Conference on Human Factors in Computing Systems (April 4-9, 2009, Boston, MA, USA), pp. 78–82.
URL: http://soft.vub.ac.be/Publications/2009/vub-prog-tr-09-03.pdf

[336] Paul Marks. *European law could limit iPod volume*. The New Scientist. Sept. 29, 2009.
URL: http://www.newscientist.com/article/dn17871

[337] Yukihiro Matsumoto et al. *Ruby (programming language)*.
URL: http://www.ruby-lang.org

[338] Ed Mayo and Tom Steinberg. *The Power of Information: An independent review*. Report commissioned by the UK government. The Cabinet Office, London, UK, 2007.
URL: http://www.opsi.gov.uk/advice/poi/index

[339] Steve McCaskill. *Smartphone Ownership In UK Reaches 50 Percent*. eWeek Europe UK, NetMediaEurope. Nov. 1, 2011.
URL: http://www.eweekeurope.co.uk/news/smartphone-ownership-in-uk-reaches-50-percent-44409

[340] Chuck Mcmanis. *The Palm V meets J2ME*. News report on the accouncement of J2ME at the 1999 JavaOne conference. JavaWorld.com. Aug. 20, 1999.
URL: http://javaworld.com/jw-08-1999/jw-08-palmv.html

[341] Tom McNichol. *The Wales Rules for Web 2.0*. CNN / Business 2.0 Magazine. 2007-07-02 2007.
URL: http://money.cnn.com/galleries/2007/biz2/0702/gallery.wikia_rules.biz2/

[342] Donella H. Meadows, Dennis L. Meadows, Jørgen Randers, and William W. Behrens III. *The Limits to Growth. A Report for The Club of Rome's Project on the Predicament of Mankind*. 1st ed. Universe Books, 1972. 205 pp.
ISBN: 0-87663-165-0

[343] Donella Meadows, Jørgen Randers, and Dennis Meadows. *Limits to Growth: The 30-Year Global Update*. Chelsea Green Publishing, 2004. 368 pp.
ISBN: 978-1931498586
URL: http://www.chelseagreen.com/bookstore/item/limitspaper

[344] Daniel Menzel, Hugo Fastl, Ralf Graf, and Jürgen Hellbrück. "Influence of vehicle color on loudness judgments". In: *Journal of the Acoustical Society of America* 123.5 (May 2008), pp. 2477–2479.
ISSN: 0001-4966
DOI: 10.1121/1.2890747

[345] Merriam-Webster, Inc. *Merriam-Webster Online Dictionary*.
URL: http://www.merriam-webster.com/dictionary

[346] Microsoft Corporation. *Windows Phone*.
URL: http://microsoft.com/windowsphone

[347] Microtech Gefell GmbH. *MK250 Kondensator Messmikrofonkapsel*. In German.
URL: http://microtechgefell.de/index.php/en/measuring-equipment/sound--noise/kapseln/87-mk250

[348] Emiliano Miluzzo et al. *CenceMe*. Dartmouth College.
URL: http://cenceme.org

[349] Emiliano Miluzzo, Nicholas D. Lane, Hong Lu, and Andrew T. Campbell. "Research in the App Store Era: Experiences from the CenceMe App Deployment on the iPhone". In: *Research in the large: Using App Stores, Markets and other wide distribution channels in UbiComp research. Workshop held at Ubicomp '10 (Copenhagen, Denmark, September 26-29, 2010)*.
URL: http://large.mobilelifecentre.org/dotclear/index.php?post/2010/05/11/Workshop-at-Ubicomp-2010

[350] Emiliano Miluzzo, Michela Papandrea, Nicholas D. Lane, Hong Lu, and Andrew T. Campbell. "Pocket, Bag, Hand, etc. - Automatically Detecting Phone Context through Discovery". In: *PhoneSense 2010: International Workshop on Sensing for App Phones (November 2, 2010), held at ACM SenSys '10 (Zurich, Switzerland, November 2-5, 2010)*, pp. 21–25.
URL: http://sensorlab.cs.dartmouth.edu/phonesense/program.html

[351] Emiliano Miluzzo et al. "Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of the CenceMe Application". In: *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems (Raleigh, NC, USA, November 4 -7, 2008)*. ACM, pp. 337–350.
ISBN: 978-1-59593-990-6
DOI: 10.1145/1460412.1460445

[352] Emiliano Miluzzo et al. "Darwin Phones: the Evolution of Sensing and Inference on Mobile Phones". In: *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services (San Francisco, CA, USA, June 15-18, 2010)*. New York, NY, USA: ACM, pp. 5–20.
ISBN: 978-1-60558-985-5
DOI: 10.1145/1814433.1814437

[353] Ministerie van de Vlaamse Gemeenschap. "22 juli 2005. — Besluit van de Vlaamse Regering inzake de evaluatie en de beheersing van het omgevingslawaai en tot wijziging van het besluit van de Vlaamse Regering van 1 juni 1995 houdende de algemene en sectorale bepalingen inzake milieuhygiëne (2005 — 2088)". In: *Moniteur Belge / Belgisch Straatsblad* 175.31.08.2005 (Aug. 2005), pp. 38020–38039. National execution measure of Directive 2002/49/EC for Flanders, Belgium.
ISSN: 0376-6764

[354] Ralph T. Muehleisen and Brent McQuillan. *Virtual Sound Level Meter*. Programmed in MATLAB and licensed under the GNU GPL version 3. 2010-2011.
URL: http://sourceforge.net/projects/vslm

[355] Min Mun et al. "PEIR, the Personal Environmental Impact Report, as a Platform for Participatory Sensing Systems Research". In: *MobiSys '09: Proceedings of the 7th international conference on Mobile systems, applications, and services (Wroclaw, Poland, June 22-25, 2009)*. New York, NY, USA: ACM, pp. 55–68.
ISBN: 978-1-60558-566-6
DOI: 10.1145/1555816.1555823

[356] Enda Murphy and Eoin A. King. "Strategic environmental noise mapping: Methodological issues concerning the implementation of the EU Environmental Noise Directive and their policy implications". Review article. In: *Environment International* 36.3 (Apr. 2010), pp. 290–298.
ISSN: 0160-4120
DOI: 10.1016/j.envint.2009.11.006

[357] Rohan Narayana Murty et al. “CitySense: An Urban-Scale Wireless Sensor Network and Testbed”. In: *Proceedings of the 2008 IEEE International Conference on Technologies for Homeland Security (Waltham, MA, USA, 12-13 May, 2008)*. IEEE, pp. 583–588.
ISBN: 978-1-4244-1977-7
DOI: 10.1109/THS.2008.4534518

[358] Sam Murugesan. “Harnessing Green IT: Principles and Practices”. In: *IT Professional* 10.1 (Jan. 2008): *Green computing*, pp. 24–33.
ISSN: 1520-9202
DOI: 10.1109/MITP.2008.10

[359] Mirco Musolesi, Mattia Piraccini, Kristof Fodor, Antonio Corradi, and Andrew T. Campbell. “Supporting Energy-Efficient Uploading Strategies for Continuous Sensing Applications on Mobile Phones”. In: *Pervasive Computing. 8th International Conference, Pervasive 2010, Helsinki, Finland, May 17-20, 2010. Proceedings.* Ed. by Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic. Lecture Notes in Computer Science, vol. 6030. Springer Berlin / Heidelberg, pp. 355–372.
ISBN: 978-3-642-12653-6
ISSN: 0302-9743
DOI: 10.1007/978-3-642-12654-3_21

[360] John Musser, Tim O’Reilly, and The O’Reilly Radar Team. *Web 2.0 Principles and Best Practices*. O’Reilly Radar Report. O’Reilly Media, Nov. 2006, p. 101.
ISBN: 0-596-52769-1
URL: http://radar.oreilly.com/research/web2-report.html

[361] mySociety Ltd. *FixMyStreet*. UK Citizens Online Democracy (UKCOD).
URL: http://www.fixmystreet.com

[362] Lama Nachman et al. “On-body health data aggregation using mobile phones”. In: *First Workshop on Sensing on Everyday Mobile Phones in Support of Participatory Research (6 November 2007), held at ACM SenSys 2007 (6–9 November 2007, Sydney, Australia)*.
URL: http://urban.cens.ucla.edu/sensys07/index.php/Accepted_Papers_and_Workshop_Agenda

[363] Bonnie A. Nardi, Diane J. Schiano, and Michelle Gumbrecht. “Blogging as Social Activity, or, Would You Let 900 Million People Read Your Diary?” In: *CSCW ’04: Proceedings of the 2004 ACM conference on Computer supported cooperative work (November 6-10, 2004; Chicago, Illinois, USA)*. New York, NY, USA: ACM, pp. 222–231.
ISBN: 1-58113-810-5
DOI: 10.1145/1031607.1031643

[364] National Audubon Society. *History of the Christmas Bird Count*.
URL: http://birds.audubon.org/history-christmas-bird-count

[365] National Institute for Occupational Safety and Health. *Criteria for a Recommended Standard: Occupational Noise Exposure (Revised Criteria 1998)*. DHHS (NIOSH) Publication No. 98-126. U.S. Department of Health, Human Services, Public Health Service, Centers for Disease Control, and Prevention, June 1998.
URL: http://www.cdc.gov/niosh/docs/98-126

[366] "The First International Acoustical Conference". In: *Nature* 140.3539 (Aug. 28, 1937), p. 370.
ISSN: 0028-0836
DOI: 10.1038/140370a0

[367] Nikolay ?Zobaken" Neizvesny. *regexp-me – Regular expressions implementation for Java ME platform*.
URL: http://code.google.com/p/regexp-me

[368] Andrew Nelson. *Travel time to major cities: A global map of Accessibility*. European Commission, Joint Research Centre. LB-30-08-671-EN-C. Luxembourg: Office for Official Publications of the European Communities, 2008.
ISBN: 978-92-79-09771-3
DOI: 10.2788/95835
URL: http://bioval.jrc.ec.europa.eu/products/gam/download.htm

[369] James Niccolai. *Oracle sues Google over Java use in Android*. IDG News Service. Aug. 12, 2010.
URL: http://www.computerworld.com/s/article/9180678

[370] Jakob Nielsen. *Participation Inequality: Encouraging More Users to Contribute*. Alertbox. Oct. 9, 2006.
ISSN: 1548-5552
URL: http://www.useit.com/alertbox/participation_inequality.html

[371] Nielsen-Kellerman. *Kestrel Wind and Weather Instruments*.
URL: http://www.nkhome.com/kestrel

[372] Nike, Inc. *Nike+*.
URL: http://nikeplus.com

[373] Nitobi Software. *PhoneGap*.
URL: http://www.phonegap.com

[374] NoiseMeters Limited. *The doseBadge Noise Dosimeter*.
URL: http://www.dosebadge.com

[375] NoiseTube Team. *NoiseTube Community Memory & project website*.
URL: http://www.noisetube.net

[376] NoiseTube Team. *NoiseTube Mobile calibration settings*. XML file with calibration settings for NoiseTube Mobile. Automatically downloaded by the app, yet human readable.
URL: noisetube.net/api/mobilecalibrations

[377] NoiseTube Team. *NoiseTube Mobile for Android*. Distributed through the Google Play service (formerly *Android Market*).
URL: https://play.google.com/store/apps/details?id=net.noisetube

[378] NoiseTube Team. *NoiseTube open source project*. Hosted by Google Code.
URL: http://code.google.com/p/noisetube

[379] NoiseTube Team. *How to calibrate a phone in 7 steps*. Instructions on how to use the now deprecrated calibration mode of NoiseTube Mobile for Java ME. 2009.
URL: http://docs.google.com/View?id=dchqdm7_298gxfb9pd9

[380] Nokia Corporation. *Java Runtime for Symbian*.
URL: http://developer.nokia.com/Community/Wiki/Category:Java_Runtime_Release_Notes

[381] Nokia Corporation. *Nokia outlines new strategy, introduces new leadership, operational structure*. Press release.
URL: http://press.nokia.com/2011/02/11/nokia-outlines-new-strategy-introduces-new-leadership-operational-structure

[382] Nokia Corporation. *Nokia Store*. Formerly called *Ovi Store*.
URL: http://store.ovi.com

[383] Nokia Corporation. *Nokia UI API*.
URL: http://developer.nokia.com/Community/Wiki/Nokia_UI_API

[384] Nokia Corporation. *Series 40 Platform (S40)*.
URL: http://developer.nokia.com/Devices/Series_40

[385] Nokia Corporation. *Nokia N95 8GB – Device details*. 2007.
URL: http://developer.nokia.com/Devices/Device_specifications/N95_8GB

[386] Nokia Corporation. *Nokia 5230 – Device details*. 2009.
URL: http://developer.nokia.com/Devices/Device_specifications/5230

[387] Nokia Corporation. *Java™ Developer's Library (v3.9)*. Aug. 12, 2011.
URL: http://developer.nokia.com/info/sw.nokia.com/id/3cfc525e-f0ec-491c-badd-085c0e2df8bf/Java_ME_Developers_Library.html

[388] Jørgen Stig Nørgård, John Peet, and Kristín Vala Ragnarsdóttir. "The History of The Limits to Growth". In: *Solutions* 1.2 (Feb. 2010), pp. 59–63.
URL: http://www.thesolutionsjournal.com/node/569

[389] Beth Simone Noveck. *Wiki Government. How Technology Can Make Government Better, Democracy Stronger, and Citizens More Powerful*. Brookings Institution Press, 2009. 224 pp.
ISBN: 978-0-8157-0510-9
URL: http://www.brookings.edu/press/Books/2009/wikigovernment.aspx

[390] Oded Nov. "What motivates Wikipedians?" In: *Communications of the ACM* 50.11 (Nov. 2007), pp. 60–64.
ISSN: 0001-0782
DOI: 10.1145/1297797.1297798

[391] Oded Nov, Ofer Arazy, and David Anderson. "Dusting for science: motivation and participation of digital citizen science volunteers". In: *iConference '11 – Proceedings of the 2011 iConference (February 08 - 11, 2011, Seattle, WA, USA)*. New York, NY, USA: ACM, pp. 68–74.
ISBN: 978-1-4503-0121-3
DOI: 10.1145/1940761.1940771

[392] Oded Nov, Ofer Arazy, and David Anderson. "Technology-Mediated Citizen Science Participation: A Motivational Model". In: *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM-11; July 17-21, 2011, Barcelona, Spain)*, pp. 249–256.
URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11

[393] Oded Nov and Chen Ye. "Why do people tag?: motivations for photo tagging". In: *Communications of the ACM* 53.7 (July 2010), pp. 128–131.
ISSN: 0001-0782
DOI: 10.1145/1785414.1785450

[394] OGC CityGML Standards Working Group. *CityGML*. Information model for the representation of 3D urban objects, built on top of the OGC Geography Markup Language 3 (GML3). Open Geospatial Consortium (OGC).
URL: http://www.citygml.org

[395] Neasan O'Neill. *World's first summit on citizen cyberscience*. Blog post announcing the London Citizen Cyberscience Summit. Citizen Cyberscience Centre. Aug. 4, 2010.
URL: http://citizencyberscience.blogspot.com/2010/08/worlds-first-summit-on-citizen.html

[396] Open Geospatial Consortium. *GeoRSS*.
URL: http://georss.org

[397] *Open Handset Alliance (OHA)*. Google-founded partnership of companies involved in the development and promotion of the Android platform.
URL: http://www.openhandsetalliance.com

[398] OpenLayers Dev Team. *OpenLayers*.
URL: http://www.openlayers.org

[399] OpenStreetMap Community and OpenStreetMap Foundation. *OpenStreetMap*.
URL: http://www.openstreetmap.org

[400] Tim Oreilly. "What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software". In: *Communications & Strategies. International journal of digital economics* 65 (First Quarter 2007): *Web 2.0 the Internet as a digital common*, pp. 17–37.
ISSN: 1157-8637
URL: http://idate.org/en/Digiworld-store/Collection/Communications-Strategies_18/No-65-Web-2-0-the-Internet-as-a-digital-common_434.html

[401] Organization for Economic Co-operation and Development. *Fighting Noise: Strengthening Noise Abatement Policies*. Development Centre Studies. OECD Publications, Oct. 1986. 145 pp.
ISBN: 978-9264128279

[402] Martin J. Osborne. *Introduction to Game Theory*. Oxford University Press, Aug. 2003. 560 pp.
ISBN: 978-0195128956
URL: http://www.economics.utoronto.ca/osborne/igt

[403] David Osimo. *Web 2.0 in Government: Why and How?* European Commission, Joint Research Centre. JRC Scientific and Technical Reports JRC 45269 / EUR 23358 EN. Office for Official Publications of the European Communities, 2008.
ISSN: 1018-5593
URL: http://ipts.jrc.ec.europa.eu/publications/pub.cfm?id=1565

[404] Elinor Ostrom. *Governing the Commons: The Evolution of Institutions for Collective Action*. Political Economy of Institutions and Decisions. Cambridge University Press, Nov. 1990. 298 pp.
ISBN: 978-0521405997
URL: http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521405997

[405] Elinor Ostrom. *Understanding Institutional Diversity*. Princeton University Press, 2005. 376 pp.
ISBN: 978-0691122380
URL: http://press.princeton.edu/titles/8085.html

[406] Elinor Ostrom. "The Challenge of Common-Pool Resources". In: *Environment: Science and Policy for Sustainable Development* 50.4 (July 2008), pp. 8–20.
ISSN: 0013-9157
DOI: 10.3200/ENVT.50.4.8-21

[407] Elinor Ostrom. "A Multi-Scale Approach to Coping with Climate Change and Other Collective Action Problems". In: *Solutions* 1.2 (Feb. 2010), pp. 27–36.
URL: http://www.thesolutionsjournal.com/node/565

[408] Elinor Ostrom, Joanna Burger, Christopher B. Field, Richard B. Norgaard, and David Policansky. "Revisiting the Commons: Local Lessons, Global Challenges". In: *Science* 284.5412 (Apr. 1999), pp. 278–282.
ISSN: 0036-8075
DOI: 10.1126/science.284.5412.278

[409] Elinor Ostrom, Roy Gardner, and James Walker. *Rules, Games, and Common-Pool Resources*. University of Michigan Press, 1994. 392 pp.
ISBN: 978-0-472-06546-2
URL: http://press.umich.edu/titleDetailLookInside.do?id=9739

[410] Adam Ostrow. *Facebook Now Has 800 Million Users*. Mashable. Sept. 22, 2011.
URL: http://mashable.com/2011/09/22/facebook-800-million-users

[411] Gwen Ottinger. "Buckets of Resistance: Standards and the Effectiveness of Citizen Science". In: *Science, Technology & Human Values* 35.2 (Mar. 2010), pp. 244–270.
ISSN: 0162-2439
DOI: 10.1177/0162243909337121

[412] Oxford University Press. *Oxford Dictionaries Online – English Dictionary and Language Reference*.
URL: http://oxforddictionaries.com

[413] Maija Palmer. *TomTom sorry for selling driver data to police*. The Financial Times. Apr. 28, 2011.
URL: http://www.ft.com/cms/s/2/3f80e432-7199-11e0-9b7a-00144feabdc0.html

[414] Tae Hong Park. *Introduction To Digital Signal Processing. Computer Musically Speaking*. World Scientific, Nov. 2009. 452 pp.
ISBN: 978-981-279-027-9
URL: http://www.worldscibooks.com/compsci/6705.html

[415] Prachi Patel-Predd. "Cellphones for Science". In: *IEEE Spectrum* 46.2 INT (Feb. 2009), p. 12.
ISSN: 0018-9235
DOI: 10.1109/MSPEC.2009.4770599

[416] Eric Paulos. "Citizen Science: Enabling Participatory Urbanism". In: *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City*. Ed. by Marcus Foth. Information Science Reference, IGI Global, 2009. Chap. 28, pp. 414–436.
ISBN: 978-1-60566-152-0
DOI: 10.4018/978-1-60566-152-0.ch028
URL: http://vrolik.de/book

[417] Eric Paulos. "Designing for Doubt: Citizen Science and the Challenge of Change". In: *Engaging Data: First International Forum on the Application and Management of Personal Electronic Information (October 12-13, 2009, Cambridge, MA, USA)*.
URL: http://senseable.mit.edu/engagingdata/papers/ED_SI_Designing_for_Doubt.pdf

[418] Eric Paulos. "You Amateur!" Feature. In: *interactions* 19.1 (Jan. 2012), pp. 52–57.
ISSN: 1072-5520
DOI: 10.1145/2065327.2065338

[419] Eric Paulos and Tom Jenkins. "Urban Probes: Encountering Our Emerging Urban Atmospheres". In: *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems (April 2-7, 2005, Portland, OR, USA)*. New York, NY, USA: ACM, pp. 341–350.
ISBN: 1-58113-998-5
DOI: 10.1145/1054972.1055020

[420] Mark Paxton. "Participate: Producing A Mass Scale Environmental Campaign for Pervasive Technology". In: *PPT&ES: Workshop on Pervasive Persuasive Technology and Environmental Sustainability (May 19, 2008), held at Pervasive 2008 (May 19-22, 2008, Sydney, Australia)*, pp. 47–50.
URL: http://www.pervasive2008.org/Papers/Workshop/w2-11.pdf

[421] Mark Paxton and Steve Benford. "Experiences of Participatory Sensing in the Wild". In: *Ubicomp '09 – Proceedings of the 11th International Conference on Ubiquitous Computing (Orlando, FL, USA, September 30 - October 3, 2009)*. New York, NY, USA: ACM, pp. 265–274.
ISBN: 978-1-60558-431-7
DOI: 10.1145/1620545.1620584

[422] Daniel Peebles, Hong Lu, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. "Community-Guided Learning: Exploiting Mobile Sensor Users to Model Human Behavior". In: *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10; Atlanta, GA, USA, July 11-15, 2010)*. AAAI Press, pp. 1600–1606.
ISBN: 978-1-57735-463-5
URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1933

[423] Phusion. *Phusion Passenger*. Also known as *mod_rails* or *mod_rack*.
URL: http://modrails.com

[424] Ken C. Pohlmann. *Principles of Digital Audio*. 6th ed. McGraw-Hill Professional, Sept. 2010. 816 pp.
ISBN: 978-0071663465
URL: http://www.mhprofessional.com/product.php?isbn=0071663460

[425] Ming-Zher Poh, Kyunghee Kim, A.D. Goessling, N.C. Swenson, and R.W. Picard. "Heartphones: Sensor Earphones and Mobile Application for Non-obtrusive Health Monitoring". In: *Proceedings of the 13th IEEE International Symposium on Wearable Computers (ISWC'09. 4-7 September 2009, Linz, Austria)*. IEEE Computer Society, pp. 153–154.
ISBN: 978-0-7695-3779-5
ISSN: 1550-4816
DOI: 10.1109/ISWC.2009.35

[426] Duncan Pollard and Rosamunde Almond, eds. *Living Planet Report 2010: Biodiversity, biocapacity and development*. World Wide Fund for Nature, Zoological Society of London and Global Footprint Network, 2010. 57 pp.
ISBN: 978-2-940443-08-6
URL: http://wwf.panda.org/about_our_earth/all_publications/living_planet_report

[427] PostgreSQL Global Development Group. *PostgreSQL*.
URL: http://www.postgresql.org

[428] Morgan Potier et al. "The Risks of Amplified Music for Disc-Jockeys Working in Nightclubs". Brief Report. In: *Ear & Hearing* 30.2 (Apr. 2009), pp. 291–293.
ISSN: 0196-0202
DOI: 10.1097/AUD.0b013e31819769fc

[429] *Personal communication with Marie Poupe (Brussels Instituut voor Milieubeheer)*. 2011.

[430] Bodhi Priyantha, Dimitrios Lymberopoulos, and Jie Liu. *LittleRock: Enabing Energy Efficient Continuous Sensing on Moble Phones*. Tech. rep. MSR-TR-2010-14. Microsoft Research, Feb. 2010. 12 pp.
URL: http://research.microsoft.com/apps/pubs/default.aspx?id=120316

[431] Bodhi Priyantha, Dimitrios Lymberopoulos, and Jie Liu. "LittleRock: Enabling Energy-Efficient Continuous Sensing on Mobile Phones". In: *IEEE Pervasive Computing* 10.2 (Apr. 2011), pp. 12–15.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2011.28

[432] J. Raddick et al. "Galaxy Zoo: An Experiment in Public Science Participation". Poster abstract. In: *Bulletin of the American Astronomical Society* 38 (Dec. 2007): *AAS Meeting Abstracts #211*. #94.03, p. 892.
ISSN: 0002-7537
URL: http://adsabs.harvard.edu/abs/2007AAS...211.9403

[433] J. Raddick et al. "Galaxy Zoo: Motivations of Citizen Scientists". Poster abstract. In: *Bulletin of the American Astronomical Society* 40 (May 2008): *AAS Meeting Abstracts #212*. #40.01, p. 240.
ISSN: 0002-7537
URL: http://adsabs.harvard.edu/abs/2008AAS...212.4001R

[434] Radio Österreich 1 (Ö1), Österreichischer Rundfunk. *Diagonal: Lärm – das Geräusch der anderen*. Radio show with associated online content (in German).
URL: http://oe1.orf.at/artikel/261460

[435] Manon Raimbault and Danièle Dubois. "Urban soundscapes: Experiences and knowledge". In: *Cities (The International Journal of Urban Policy and Planning)* 22.5 (Oct. 2005), pp. 339–350.
ISSN: 0264-2751
DOI: 10.1016/j.cities.2005.05.003

[436] Manon Raimbault, Catherine Lavandier, and Michel Berengier. "Ambient sound assessment of urban environments: field studies in two French cities". In: *Applied Acoustics* 64.12 (Dec. 2003), pp. 1241–1256.
DOI: 10.1016/S0003-682X(03)00061-6

[437] Rajib Kumar Rana, Chun Tung Chou, and Salil Kanhere. "Reconstruction of temporal-spatial profile from participatory sensing data using Compressive Sensing". In: *First Workshop on Sensing on Everyday Mobile Phones in Support of Participatory Research (6 November 2007), held at ACM SenSys 2007 (6–9 November 2007, Sydney, Australia)*.
URL: http://urban.cens.ucla.edu/sensys07/index.php/Accepted_Papers_and_Workshop_Agenda

[438] Rajib Kumar Rana, Chun Tung Chou, Salil S. Kanhere, Nirupama Bulusu, and Wen Hu. "Ear-Phone: An End-to-End Participatory Urban Noise Mapping System". In: *IPSN 2010: Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (April 12-16, 2010, Stockholm, Sweden)*. New York, NY, USA: ACM, pp. 105–116.
ISBN: 978-1-60558-988-6
DOI: 10.1145/1791212.1791226

[439] Sasank Reddy, Deborah Estrin, Mark Hansen, and Mani Srivastava. "Examining Micro-Payments for Participatory Sensing Data Collections". In: *Ubicomp '10: Proceedings of the 12th ACM international conference on Ubiquitous computing (Copenhagen, Denmark, September 26-29, 2010)*. New York, NY, USA: ACM, pp. 33–36.
ISBN: 978-1-60558-843-8
DOI: 10.1145/1864349.1864355

[440] Sasank Reddy, Deborah Estrin, and Mani Srivastava. "Recruitment Framework for Participatory Sensing Data Collections". In: *Pervasive Computing. 8th International Conference, Pervasive 2010, Helsinki, Finland, May 17-20, 2010. Proceedings*. Ed. by Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic. Lecture Notes in Computer Science, vol. 6030. Springer Berlin / Heidelberg, pp. 138–155.
ISBN: 978-3-642-12653-6
ISSN: 0302-9743
DOI: 10.1007/978-3-642-12654-3_9

[441] Sasank Reddy et al. "Evaluating Participation and Performance in Participatory Sensing". In: *UrbanSense08: Proceedings of International Workshop on Urban, Community, and Social Applications of Networked Sensing Systems (November 4, 2008; Raleigh, NC, USA)*. Held in conjunction with ACM SenSys 2008 (Raleigh, NC, USA, November 4 -7, 2008), pp. 1–5.
URL: http://sensorlab.cs.dartmouth.edu/urbansensing/papers/reddy_urbansense08.pdf

[442] Sasank Reddy et al. "Using Context Annotated Mobility Profiles to Recruit Data Collectors in Participatory Sensing". In: *Location and Context Awareness, 4th International Symposium, LoCA 2009 Tokyo, Japan, May 7-8, 2009. Proceedings*. Ed. by Tanzeem Choudhury, Aaron Quigley, Thomas Strang, and Koji Suginuma. Lecture Notes in Computer Science, vol. 5561. Springer Berlin / Heidelberg, pp. 52–69.
ISBN: 978-3-642-01720-9
ISSN: 0302-9743
DOI: 10.1007/978-3-642-01721-6_4

[443] Sasank Reddy et al. "Biketastic: Sensing and Mapping for Better Biking". In: *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems (Atlanta, GA, USA, April 10-15, 2010)*. New York, NY, USA: ACM, pp. 1817–1820.
ISBN: 978-1-60558-929-9
DOI: 10.1145/1753326.1753598

[444] Sasank Reddy et al. "Using Mobile Phones to Determine Transportation Modes". In: *ACM Transactions on Sensor Networks* 6.2 (Feb. 2010). Article No. 13, p. 27.
ISSN: 1550-4859
DOI: 10.1145/1689239.1689243

[445] Red Hat, Inc. and GNU Classpath. *IcedTea*.
URL: http://icedtea.classpath.org

[446] Trygve Reenskaug. "The Model-View-Controller (MVC). Its Past and Present". In: *JavaZONE (Oslo, Norway, 18-19 September 2003)*.
URL: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html

[447] Refractions Research and third party contributors. *PostGIS*.
URL: http://postgis.refractions.net

[448] Jun Rekimoto, Takashi Miyaki, and Takaaki Ishizawa. "LifeTag: WiFi-Based Continuous Location Logging for Life Pattern Analysis". In: *Location- and Context-Awareness, Third International Symposium, LoCA 2007, Oberpfaffenhofen, Germany, September 20-21, 2007. Proceedings*. Ed. by Jeffrey Hightower, Bernt Schiele, and Thomas Strang. Lecture Notes in Computer Science, vol. 4718. Springer Berlin / Heidelberg, pp. 35–49.
ISBN: 978-3-540-75159-5
ISSN: 0302-9743
DOI: 10.1007/978-3-540-75160-1_3

[449] Franklin Reynolds. "Camera Phones: A Snapshot of Research and Applications". In: *IEEE Pervasive Computing* 7.2 (Apr. 2008), pp. 16–19.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2008.28

[450] Howard Rheingold. *Smart Mobs: The Next Social Revolution*. Basic Books, 2002. 288 pp.
ISBN: 978-0738206080
URL: http://www.smartmobs.com

[451] George Ritzer and Nathan Jurgenson. "Production, Consumption, Prosumption". In: *Journal of Consumer Culture* 10.1 (Mar. 2010), pp. 13–36.
ISSN: 1469-5405
DOI: 10.1177/1469540509354673

[452] Hasan Rizvi and Jeet Kaul. *Oracle + Sun: Java Strategy*. Video webcast from Oracle Corporation. Jan. 2010.
URL: http://oracle.com.edgesuite.net/ivt/4000/8104/9236/12630/lobby_external_flash_clean_480x360/default.htm

[453] Kay Römer and Friedmann Mattern. "The Design Space of Wireless Sensor Networks". In: *IEEE Wireless Communications* 11.6 (Dec. 2004), pp. 54–61.
ISSN: 1536-1284
DOI: 10.1109/MWC.2004.1368897

[454] Michael Rossman. "Implications of Community Memory". In: *ACM SIGCAS Computers and Society* 6.4 (Winter 1975), pp. 7–10.
ISSN: 0095-2737
DOI: 10.1145/958785.958789

[455] Philip E. Ross. "Top 11 technologies of the decade". In: *IEEE Spectrum* 48.1 INT (Jan. 2011), pp. 23–59.
ISSN: 0018-9235
DOI: 10.1109/MSPEC.2011.5676379

[456] Royal Swedish Academy of Sciences, Economic Sciences Prize Committee. *Scientific Background on the Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel 2009: Economic Governance*. Oct. 2009. 19 pp.
URL: http://nobelprize.org/nobel_prizes/economics/laureates/2009

[457] David Sang. *Cambridge IGCSE Physics Coursebook*. Cambridge University Press, 2010. 288 pp.
ISBN: 978-0521757737
URL: http://www.cambridge.org/education/international/subject/project/item5629552

[458] Silvia Santini. "Adaptive Sensor Selection Algorithms for Wireless Sensor Networks". PhD thesis. Zurich, Switzerland: ETH Zurich, Oct. 2009. 231 pp.
URL: http://www.inf.ethz.ch/personal/santinis/publications

[459] Silvia Santini, Benedikt Ostermaier, and Robert Adelmann. "On the Use of Sensor Nodes and Mobile Phones for the Assessment of Noise Pollution Levels in Urban Environments". In: *INSS2009: Sixth International Conference on Networked Sensing Systems (17-19 June, 2009, Pittsburgh, PA, USA)*. IEEE, pp. 1–8.
ISBN: 978-1-4244-6313-8
DOI: 10.1109/INSS.2009.5409957

[460] Silvia Santini, Benedikt Ostermaier, and Andrea Vitaletti. "First Experiences Using Wireless Sensor Networks for Noise Pollution Monitoring". In: *REALWSN '08: Proceedings of the 3rd ACM Workshop on Real-World Wireless Sensor Networks (April 1, 2008; Glasgow, UK)*. New York, NY, USA: ACM, pp. 61–65.
ISBN: 978-1-60558-123-1
DOI: 10.1145/1435473.1435490

[461] Joshua Schachter, Yahoo!, Inc., et al. *Delicious*. Previously called del.icio.us. AVOS Systems, Inc.
URL: http://delicious.com

[462] R. Murray Schafer. *The New Soundscape: A handbook for the modern music teacher*. BMI Canada, 1969. 65 pp.
ISBN: 978-0911320916

[463] Joke Schauvliege. *Regeling geluidsnormen voor muziekactiviteiten*. Press release and Q&A document; in Dutch. Flemish Ministry of Environment, Nature and Culture. June 10, 2011.
URL: http://www.lne.be/themas/hinder-en-risicos/geluidshinder/beleid/muziek

[464] Doug Schuler. "Community Networks: Building a New Participatory Medium". In: *Communications of the ACM* 37.1 (Jan. 1994), pp. 38–51.
ISSN: 0001-0782
DOI: 10.1145/175222.175225

[465] Hillel Schwartz. *Making Noise: From Babel to the Big Bang and Beyond*. Zone Books / MIT Press, Oct. 2011. 928 pp.
ISBN: 978-1-935408-12-3
URL: http://www.zonebooks.org/titles/SCHW_MAK.html

[466] Immanuel Schweizer et al. *da_sense – Darmstadt Sensor Netzwerk*. Technische Universität Darmstadt. 2011.
URL: http://www.da-sense.de

[467] Immanuel Schweizer, Roman Bärtl, Axel Schulz, Forian Probst, and Max Mühläuser. "NoiseMap - Real-time participatory noise maps". In: *PhoneSense 2011 – Second International Workshop on Sensing Applications on Mobile Phones (November 1, 2011), held at ACM SenSys 2011 (November 1-4, 2011, Seattle, WA, USA)*.
URL: http://research.microsoft.com/en-us/um/redmond/events/phonesense2011/program.htm

[468] John P. Seiler. "A Historical Perspective on the Evaluation, Standardization and Certification of Personal Noise Dosimeters". In: *The 2008 NIOSH Direct-Reading Exposure Assessment Methods (DREAM) Workshop (November 13-14, 2008; Washington DC, USA)*.
URL: http://www.aiha.org/insideaiha/documents/Seiler.pdf

[469] Eberhard Sengpiel. *Correlation between sones und phons ? calculator*.
URL: http://www.sengpielaudio.com/calculatorSonephon.htm

[470] Sensaris. *Senspod*.
URL: http://www.sensaris.com/products/senspod

[471] Stephen Shankland. *Google's Android parts ways with Java industry group*. CNET News. Nov. 12, 2007.
URL: http://news.cnet.com/8301-13580_3-9815495-39.html

[472] Dave Shapton. "Digital Microphones: A New Approach?" Cutting Edge. In: *Sound On Sound* (Mar. 2004).
URL: http://www.soundonsound.com/sos/mar04/articles/cuttingedge.htm

[473] Shenzhen Municipal Science and Technology Co., Ltd. *CEM DT-8852*.
URL: http://www.cem-hh.com/en/DT-8852.php

[474] Jing Shi, Rui Zhang, Yunzhong Liu, and Yanchao Zhang. "PriSense: Privacy-Preserving Data Aggregation in People-Centric Urban Sensing Systems". In: *Proceedings of IEEE INFOCOM 2010 (14-19 March, 2010, San Diego, CA)*. IEEE, pp. 1–9.
ISBN: 978-1-4244-5836-3
ISSN: 0743-166X
DOI: 10.1109/INFCOM.2010.5462147

[475] Katie Shilton. "Four Billion Little Brothers?: Privacy, mobile phones, and ubiquitous data collection". In: *Communications of the ACM* 52.11 (Nov. 2009), pp. 48–53.
ISSN: 0001-0782
DOI: 10.1145/1592761.1592778

[476] Katie Shilton, Jeff Burke, Deborah Estrin, Mark Hansen, and Mani B. Srivastava. "Participatory Privacy in Urban Sensing". In: *Proceedings of the International Workshop on Mobile Device and Urban Sensing (MODUS; St. Louis, MO, USA, April 21, 2008)*.
URL: http://escholarship.org/uc/item/90j149pp

[477] Michael Shirts and Vijay S. Pande. "Screen Savers of the World Unite!" Perspective. In: *Science* 290.5498 (Dec. 8, 2000), pp. 1903–1904.
ISSN: 0036-8075
DOI: 10.1126/science.290.5498.1903

[478] Irina Shklovski and Michele F. Chang. "Urban Computing: Navigating Space and Context". Guest Editors' Introduction. In: *Computer* 39.9 (Sept. 2006), pp. 36–37.
ISSN: 0018-9162
DOI: 10.1109/MC.2006.308

[479] Shockfish SA. *Tinynode*.
URL: http://www.tinynode.com

[480] Jonathan Silvertown. "A new dawn for citizen science". Update. In: *Trends in Ecology & Evolution* 24.9 (Sept. 2009), pp. 467–471.
ISSN: 0169-5347
DOI: 10.1016/j.tree.2009.03.017

[481] Rytis Slatkevi?ius and PrimeGrid community. *PrimeGrid project*. 2005.
URL: http://www.primegrid.com

[482] Joyce Slaton. *Remembering Community Memory – The Berkeley beginnings of online community*. SFGate. Dec. 13, 2001.
URL: http://www.sfgate.com/gate/archive/2001/12/13/commmem.DTL

[483] Brett Smith. *FSF responds to Oracle v. Google and the threat of software patents*. Free Software Foundation. Sept. 8, 2010.
URL: http://www.fsf.org/news/oracle-v-google

[484] Kazem Sohraby, Daniel Minoli, and Taieb Znati. *Wireless Sensor Networks: Technology, Protocols, and Applications*. Wiley, May 2007. 328 pp.
ISBN: 978-0-471-74300-2
URL: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0471743003.html

[485] Martin Sonn and Acoustical Society of America. *American National Standard Psychoacoustical Terminology*. ANSI/ASA S3.20-1973. American National Standards Institute, Dec. 1973. 67 pp. Withdrawn, replaced by ANSI/ASA S3.20-1995.

[486] Sony Ericsson. *W995. Seamless music experience*. White paper 1220.9550 R1A. Feb. 2009. 59 pp.
URL: http://developer.sonyericsson.com

[487] Sony Ericsson Mobile Communications AB. *Developers guidelines. Java™ Platform, Micro Edition, CLDC – MIDP 2 for Sony Ericsson feature and entry level phones*. Version R36A. June 2010.
URL: http://developer.sonyericsson.com/wportal/devworld/downloads/download/dw-65067-dgjavamer34a

[488] "Reken- en meetvoorschift geluidhinder 2006". VROM. In: *Staatscourant* 2006.249 (Dec. 21, 2006), p. 84. Official Dutch calculation and measuring regulation for environmental noise. Aligned to the requirements of the EU's Environmental Noise Directive.
URL: http://www.stillerverkeer.nl/index.php?page=downloadsRMV

[489] Stad Leuven. *Gecoördineerd politiereglement van de stad Leuven*. Municipal police regulations of the city of Leuven, Belgium; in Dutch. Aug. 1, 2005.
URL: http://www.lokalepolitie.be/5388/politiereglement.html

[490] D. A. Stainforth et al. "Uncertainty in predictions of the climate response to rising levels of greenhouse gases". Letter. In: *Nature* 433.7024 (Jan. 2005), pp. 403–406.
ISSN: 0028-0836
DOI: 10.1038/nature03301

[491] Luc Steels. "Expert Systems And Beyond: Community Memories". In: *Expert Systems and Knowledge Engineering: Essential Elements of Advanced Information Technology. Proceedings of the Technology Assessment and Management Conference of the Gottlieb Duttweiler Institute (Rüschlikon, Zürich, Switzerland, April 25-26, 1985)*. Ed. by Thomas Bernold. Elsevier Science Publishers (North-Holland), Apr. 1986, pp. 17–29.
ISBN: 0-444-70057-9

[492] Luc Steels. "Collaborative tagging as distributed cognition". In: *Pragmatics & Cognition* 14.2 (2006), pp. 287–292.
ISSN: 0929-0907
DOI: 10.1075/pc.14.2.09ste

[493] Luc Steels. *Community Memories for Sustainable Societies*. To be published on the occasion of the 20th Anniversary of Sony CSL, Tokyo, spring 2008. Sony Computer Science Laboratory – Paris, Nov. 2007.
URL: http://www.csl.sony.fr/wp-content/uploads/2007/11/anniv-steels1.pdf

[494] Luc Steels and Eugenio Tisseli. "Social Tagging in Community Memories". In: *Social Information Processing - Papers from the 2008 AAAI Spring Symposium (March 26-28, 2008; Stanford University)*. Technical Report SS-08-06. Menlo Park, California, USA: AAAI Press, pp. 98–103.
URL: http://www.aaai.org/Library/Symposia/Spring/ss08-06.php

[495] Luc Steels and Eugenio Tisselli. "Interfaces for Community Memories". In: *Proceedings of the Intelligent User Interfaces for Developing Regions workshop (IUI4DR), held at IUI 2008, the 13th international conference on Intelligent User Interfaces (January 13-16, 2008, Gran Canaria, Spain)*, pp. 23–28.
URL: http://www.uni-koblenz.de/~confsec/IUI/proceedings/data/workshops.html

[496] Matthias Stevens. *NoiseTube User Guide*. v1.02. NoiseTube Team. Nov. 2009. Not up-to-date with respect to current Java ME app version; does not cover the Android and iOS apps.
URL: http://noisetube.net/help

[497] Matthias Stevens. *Motivations for reusing and sharing code between the NoiseTube Mobile apps for Java ME and Android*. Note posted with NoiseTube Mobile source code. NoiseTube Team. Aug. 2011.
URL: http://noisetube.googlecode.com/svn/mobile/java/shared/Motivations.pdf

[498] Matthias Stevens and Ellie D'Hondt. "Participatory Sensing for Sustainable Urban Life". Poster displayed at the MoVES project annual meeting (Université catholique de Louvain, Louvain-la-Neuve, Belgium, December 17, 2009).
URL: http://soft.vub.ac.be/Publications/2009/vub-soft-tr-09-05.pdf

[499] Matthias Stevens and Ellie D'Hondt. "Crowdsourcing of Pollution Data using Smartphones". In: *Proceedings of the Workshop on Ubiquitous Crowdsourcing, held at Ubicomp '10 (Copenhagen, Denmark, September 26-29, 2010)*.
URL: http://soft.vub.ac.be/Publications/2010/vub-tr-soft-10-15.pdf

[500] Matthias Stevens and Ellie D'Hondt. "NoiseTube & BrusSense: Participatory sensing for sustainable urban life". In: *We Can Change the Weather: 100 Cases in Energy Efficacy to Start Doing So*. Ed. by Marleen Wynants and Sara Engelen. Crosstalks. ASP-VUBPress, 2010, pp. 106–107.
ISBN: 978-9054876922
URL: http://crosstalks.vub.ac.be/publications/changetheweather/intro.html

[501] Matthias Stevens and Ellie D'Hondt. "Participatory Sensing for Sustainable Urban Life". Poster displayed at the Dag van de Doctorandi event (VUB, Brussels, Belgium, May 28, 2010).
URL: http://soft.vub.ac.be/Publications/2009/vub-soft-tr-09-05.pdf

[502] Stanley Smith Stevens. "A scale for the measurement of a psychological magnitude: loudness". In: *Psychological Review* 43.5 (Sept. 1936), pp. 405–416.
ISSN: 0033-295X
DOI: 10.1037/h0058773

[503] Wayne P. Stevens, Glenford J. Myers, and Larry L. Constantine. "Structured design". In: *IBM Systems Journal* 13.2 (June 1974), pp. 115–139.
ISSN: 0018-8670
DOI: 10.1147/sj.132.0115

[504] Richard L. St. Pierre Jr. and Daniel J. Maguire. "The Impact of A-Weighting Sound Pressure Level Measurements during the Evaluation of Noise Exposure". In: *NOISE-CON 2004 – The National Conference on Noise Control Engineering (July 12-14, 2004; Baltimore, MD, USA)*, pp. 702–708.
URL: http://trid.trb.org/view.aspx?id=813685

[505] Sujesha Sudevalayam and Purushottam Kulkarni. "Energy Harvesting Sensor Nodes: Survey and Implications". In: *IEEE Communications Surveys Tutorials* 13.3 (Third quarter 2011), pp. 443–461.
ISSN: 1553-877X
DOI: 10.1109/SURV.2011.060710.00094

[506] Brian L. Sullivan et al. "eBird: A citizen-based bird observation network in the biological sciences". In: *Biological Conservation* 142.10 (Oct. 2009), pp. 2282–2292.
ISSN: 0006-3207
DOI: 10.1016/j.biocon.2009.05.006

[507] Sun Microsystems, Inc. and Oracle Corporation. *About Java ME*.
URL: http://www.oracle.com/technetwork/java/javame/about-java-me-395899.html

[508] Sun Microsystems, Inc. and Oracle Corporation. *Connected Device Configuration (CDC)*.
URL: http://www.oracle.com/technetwork/java/javame/tech/index-jsp-139293.html

[509] Sun Microsystems, Inc. and Oracle Corporation. *Connected Limited Device Configuration (CLDC)*.
URL: http://java.sun.com/products/cldc

[510] Sun Microsystems, Inc. and Oracle Corporation. *HotSpot*. Java Virtual Machine.
URL: http://java.sun.com/products/hotspot

[511] Sun Microsystems, Inc. and Oracle Corporation. *Java ME Software Development Kit (SDK)*.
URL: http://java.sun.com/javame/sdk

[512] Sun Microsystems, Inc. and Oracle Corporation. *Java Platform, Enterprise Edition (Java EE)*. Formerly known as "Java 2 Platform, Enterprise Edition" (J2EE).
URL: http://www.oracle.com/technetwork/java/javaee

[513] Sun Microsystems, Inc. and Oracle Corporation. *Java Platform, Micro Edition (Java ME)*. Formerly known as "Java 2 Platform, Micro Edition" (J2ME).
URL: http://www.oracle.com/technetwork/java/javame

[514] Sun Microsystems, Inc. and Oracle Corporation. *Java Platform, Standard Edition (Java SE)*. Formerly known as "Java 2 Platform, Standard Edition" (J2SE).
URL: http://www.oracle.com/technetwork/java/javase

[515] Sun Microsystems, Inc. and Oracle Corporation. *Java (programming language)*.
URL: http://www.oracle.com/technetwork/java

[516] Sun Microsystems, Inc. and Oracle Corporation. *Lightweight User Interface Toolkit (LWUIT)*.
URL: http://lwuit.java.net

[517] Sun Microsystems, Inc. and Oracle Corporation. *Mobile Information Device Profile (MIDP)*.
URL: http://java.sun.com/products/midp

[518] Sun Microsystems, Inc., Oracle Corporation, and JCP members. *Java Community Process (JCP) Program*.
URL: http://jcp.org

[519] Symbian, Ltd., Symbian Foundation, Ltd., and Nokia Corporation. *Symbian*. Mobile operating system and computing platform.
URL: http://symbian.nokia.com

[520] Igor Sysoev et al. *nginx*. Open source HTTP server.
URL: http://nginx.org

[521] Mark Szpakowski. *Community Memory – 1972-1974, Berkeley and San Francisco, California*. Community Memory Project. 1997–2006.
URL: http://www.well.com/~szpak/cm

[522] TAGora consortium. *TAGora – Semiotic Dynamics in Online Social Communities*. European project (FP6/STREP). 2006–2009.
URL: http://www.tagora-project.eu

[523] Don Tapscott and Anthony D. Williams. *Wikinomics: How Mass Collaboration Changes Everything*. Portfolio, 2006. 324 pp.
ISBN: 978-1591841388
URL: http://www.wikinomics.com/book

[524] *The Food Trust*.
URL: http://www.thefoodtrust.org

[525] Arvind Thiagarajan et al. “VTrack: Accurate, Energy-aware Road Traffic Delay Estimation Using Mobile Phones”. In: *SenSys '09: Proceedings of the 7th ACM Conference on Embedded Network Sensor Systems (Berkeley, CA, USA, November 4-6, 2009)*. New York, NY, USA: ACM, pp. 85–98.
ISBN: 978-1-60558-519-2
DOI: 10.1145/1644038.1644048

[526] Tides Center. *Global Community Monitor*.
URL: http://www.gcmonitor.org

[527] TinyOS Alliance. *TinyOS*. 2000-2010.
URL: http://www.tinyos.net

[528] Eugenio Tisselli et al. *megafone.net*. Formerly known as *zexe.net*. 2003–2011.
URL: http://megafone.net

[529] Eugenio Tisselli. “thinkflickrthink: A Case Study on Strategic Tagging”. In: *Communications of the ACM* 53.8 (Aug. 2010), pp. 141–145.
ISSN: 0001-0782
DOI: 10.1145/1787234.1787270

[530] TomTom International BV. *Live Traffic*.
URL: http://www.tomtom.com/livetraffic

[531] Linus Torvalds et al. *Linux kernel*. Linux Kernel Organization, Inc.
URL: http://www.kernel.org

[532] Graham M. Turner. “A comparison of *The Limits to Growth* with 30 years of reality”. In: *Global Environmental Change* 18.3 (Aug. 2008): *Globalisation and Environmental Governance: Is Another World Possible?*, pp. 397–411.
ISSN: 0959-3780
DOI: 10.1016/j.gloenvcha.2008.05.001

[533] UC Berkeley College of Engineering. *Mobile Millennium project*. In partnership with CCIT, Caltrans, Nokia and NAVTEQ.
URL: http://traffic.berkeley.edu

[534] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. *The Anatomy of the Facebook Social Graph*. Preprint article. Nov. 18, 2011.
URL: http://arxiv.org/abs/1111.4503

[535] UK Government. *Data.gov.uk*.
URL: http://data.gov.uk

[536] United Nations. *Rio+20 – United Nations Conference on Sustainable Development (Rio de Janeiro, Brazil; June 4-6, 2012)*.
URL: http://www.uncsd2012.org

[537] United Nations, Department of Economic and Social Affairs (UN DESA), Population Division. *World Urbanization Prospects: The 2005 Revision*. 2005.
URL: http://www.un.org/esa/population/publications/WUP2005/2005wup.htm

[538] United Nations, Department of Economic and Social Affairs (UN DESA), Population Division, Population Estimates and Projections Section. *World Population Prospects, the 2010 Revision*. May 2011.
URL: http://esa.un.org/unpd/wpp

[539] United Nations Environment Programme. *Global Environment Outlook 3. Past, present and future perspectives*. 2002. 480 pp.
ISBN: 92-807-2088-0
ISSN: 1366-8080
URL: http://www.unep.org/geo/geo3

[540] United Nations Environment Programme (UNEP). *Rio Declaration on Environment and Development*. Proclaimed at the United Nations Conference on Environment and Development (Rio de Janeiro, Brasil, 3-14 June 1992). 1992.
URL: http://www.unep.org/Documents.Multilingual/Default.asp?DocumentID=78&ArticleID=1163

[541] University of California, Los Angeles (UCLA). *Center for Embedded Networked Sensing (CENS)*.
URL: http://research.cens.ucla.edu

[542] University of Cambridge. *Cambridge Mobile Urban Sensing (CamMobSens)*.
URL: http://www.escience.cam.ac.uk/mobiledata

[543] United Nations. *Kyoto Protocol to the United Nations Framework Convention on Climate Change*. Dec. 11, 1997. 20 pp.
URL: http://unfccc.int/kyoto_protocol/items/2830.php

[544] U.S. Department of Labor, Occupational Safety & Health Administration. "Occupational Noise Exposure: Hearing Conservation Amendment; Final Rule". In: *Federal Register* 48.46 (1983), pp. 9738–9785. Adopted as standard OSHA 29 CFR 1910.95.
URL: http://osha.gov/pls/oshaweb/owadisp.show_document?p_table=STANDARDS&p_id=9735

[545] U.S. Energy Information Administration. *International Energy Outlook 2011*. Report DOE/EIA-0484(2011). Sept. 2011. 301 pp.
URL: http://www.eia.gov/forecasts/ieo

[546] U.S. government. *Data.gov*.
URL: http://www.data.gov

[547] Jorge Vallejos, Matthias Stevens, Ellie D'Hondt, Nicolas Maisonneuve, Wolfgang De Meuter, Theo D'Hondt, and Luc Steels. "Context-aware Resource Sharing for People-centric Sensing". In: *First International Workshop on Software Research and Climate Change (WSRCC-1), part of Onward! 2009 (co-located with OOPSLA 2009) in Orlando, FL, USA, October 26, 2009*. Ed. by Steve Easterbrook.
URL: http://www.cs.toronto.edu/wsrcc/WSRCC1/Papers.html

[548] Tom Van Cutsem, Stijn Mostinckx, Elisa Gonzalez Boix, Jessie Dedecker, and Wolfgang De Meuter. "AmbientTalk: object-oriented event-driven programming in Mobile Ad hoc Networks". In: *Proceedings of the XXVI International Conference of the Chilean Society of Computer Science (SCCC 2007; November 8-9, 2007, Iquique, Chili)*. Ed. by Hernán Astudillo and Éric Tanter. IEEE Computer Society, pp. 3–12.
ISBN: 0-7695-3017-6
ISSN: 1522-4902
DOI: 10.1109/SCCC.2007.12

[549] Marjan van den Belt. *Mediated Modeling. A System Dynamics Approach To Environmental Consensus Building*. Island Press, 2004. 296 pp.
ISBN: 978-1559639613
URL: http://islandpress.org/bookstore/details6619.html?prod_id=997

[550] Steve Vanlanduit and Marc Van Overmeire. *Akoestiek en geluidshinder*. Syllabus on acoustics and noise nuisance (in Dutch). Vrije Universiteit Brussel, 2008. 166 pp.

[551] Timothy Van Renterghem, Pieter Thomas, Frederico Dominguez, Samuel Dauwe, Abdellah Touhafi, Bart Dhoedt, and Dick Botteldooren. "On the ability of consumer electronics microphones for environmental noise monitoring". In: *Journal of Environmental Monitoring* 13.3 (Mar. 2011), pp. 544–552.
ISSN: 1464-0325
DOI: 10.1039/C0EM00532K

[552] Mac E. Van Valkenburg and Wendy M. Middleton, eds. *Reference Data for Engineers: Radio, Electronics, Computers and Communications*. 9th ed. Newnes, Oct. 2001. 1672 pp.
ISBN: 978-0-7506-7291-7
URL: http://www.elsevierdirect.com/product.jsp?isbn=9780750672917

[553] John Vincke. *Sociologie. Een klassieke en hedendaagse benadering*. In Dutch. Academia Press, 2007. 350 pp.
ISBN: 978-9038211046

[554] VisionMobile Ltd. *Mobile Developer Economics 2010 and Beyond – Making sense of a fragmented world*. Market research report. July 2010. 57 pp.
URL: http://www.visionmobile.com/devecon.php

[555] VisionMobile Ltd. *Developer Economics 2011 – How developers and brands are making money in the mobile app economy*. Market research report. June 2011. 60 pp.
URL: http://www.visionmobile.com/devecon.php

[556] Vlaamse overheid, Departement Leefmilieu, Natuur en Energie (LNE). *Richtlijn Omgevingslawaai – geluidskaarten en actieplannen in Vlaanderen*. Brochure. Sept. 2009. 56 pp. In Dutch.
URL: http://www.lne.be/themas/hinder-en-risicos/geluidshinder/beleid/eu-richtlijn/brochure-richtlijn-omgevingslawaai/brochure-richtlijn-omgevingslawaai

[557] Vlaamse overheid, Departement Leefmilieu, Natuur en Energie (LNE). *Goedgekeurde geluidskaarten*. Noise maps by the Flemish government. In Dutch. 2009–2012.
URL: http://www.lne.be/themas/hinder-en-risicos/geluidshinder/beleid/eu-richtlijn/goedgekeurde-geluidskaarten

[558] Luis von Ahn and Laura Dabbish. "Labeling Images with a Computer Game". In: *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems (April 24-29, 2004, Vienna, Austria)*. New York, NY, USA: ACM, pp. 319–326.
ISBN: 1-58113-702-8
DOI: 10.1145/985692.985733

[559] Luis von Ahn and Laura Dabbish. “Designing Games With A Purpose”. In: *Communications of the ACM* 51.8 (Aug. 2008), pp. 58–67.
ISSN: 0001-0782
DOI: 10.1145/1378704.1378719

[560] Mathis Wackernagel et al. “Tracking the ecological overshoot of the human economy”. In: *Proceedings of the National Academy of Sciences (PNAS)* 99.14 (July 2002), pp. 9266–9271.
ISSN: 0027-8424
DOI: 10.1073/pnas.142033699

[561] The Wall Street Journal. *Full Text: Nokia CEO Stephen Elop’s ‘Burning Platform’ Memo*. Feb. 9, 2011.
URL: http://on.wsj.com/hHJkHF

[562] Yi Wang et al. “A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition”. In: *MobiSys ’09: Proceedings of the 7th international conference on Mobile systems, applications, and services (Wroclaw, Poland, June 22-25, 2009)*. New York, NY, USA: ACM, pp. 179–192.
ISBN: 978-1-60558-566-6
DOI: 10.1145/1555816.1555835

[563] Dennis Ward, K. Meymaris, S. Henderson, and R. Johnson. “Starry Nights: The Great World Wide Star Count”. In: *Bulletin of the American Astronomical Society* 40 (May 2008): *AAS Meeting Abstracts #212*. #50.02, p. 245.
ISSN: 0002-7537
URL: http://adsabs.harvard.edu/abs/2008AAS...212.4001R

[564] Richard Watson and Owen Downey. *The Little Red Book of Acoustics. A Practical Guide*. 2nd ed. Blue Tree Acoustics, July 2008. 274 pp.
ISBN: 978-0956001207
URL: http://www.bluetreeacoustics.co.uk/books.html

[565] Weather Underground, Inc. *Historical weather conditions in Antwerp, Beligum*. Dec. 9, 2010.
URL: http://wunderground.com/history/airport/EBAW/2010/12/9/DailyHistory.html

[566] Mark Weiser. “The Computer for the 21st Century”. In: *Scientific American* 265.3 (Sept. 1991), pp. 94–104.
ISSN: 0036-8733

[567] Mark Weiser. “Hot topics - Ubiquitous Computing”. In: *Computer* 26.10 (Oct. 1993), pp. 71–72.
ISSN: 0018-9162
DOI: 10.1109/2.237456

[568] Mark Weiser. "Some computer science issues in ubiquitous computing". In: *Communications of the ACM* 36.7 (July 1993), pp. 75–84.
ISSN: 0001-0782
DOI: 10.1145/159544.159617

[569] Aaron Weiss. "The Power of Collective Intelligence". In: *netWorker* 9.3 (Sept. 2005): *Beyond file-sharing: collective intelligence*, pp. 16–23.
ISSN: 1091-3556
DOI: 10.1145/1086762.1086763

[570] WHO Regional Office for Europe. *Night noise guidelines for Europe*. Tech. rep. Scherfigsvej 8, DK-2100 Copenhagen Ø, Denmark, 2009. 184 pp.
ISBN: 978-92-890-4173-7
URL: http://www.euro.who.int/en/what-we-do/health-topics/environmental-health/noise/publications/2009/night-noise-guidelines-for-europe

[571] WHO Regional Office for Europe. *Health and Environment in Europe: Progress Assessment*. Ed. by Malcolm MacGarvin and Rosemary Bohr. 2010. 168 pp.
ISBN: 978-92-890-41680
URL: http://www.euro.who.int/en/what-we-publish/abstracts/health-and-environment-in-europe-progress-assessment

[572] WHO Regional Office for Europe / European Commission Joint Research Centre. *Burden of disease from environmental noise. Quantification of healthy life years lost in Europe*. 2011. 126 pp.
ISBN: 978-92-890-0229-5
URL: http://www.euro.who.int/en/what-we-do/health-topics/environment-and-health/noise/publications/2011/burden-of-disease-from-environmental-noise.-quantification-of-healthy-life-years-lost-in-europe

[573] WideTag, Inc. *WideNoise*. 2009.
URL: http://www.widetag.com/widenoise

[574] Wikipedia. *Adaptive Multi-Rate audio codec (AMR)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Adaptive_Multi-Rate_audio_codec

[575] Wikipedia. *Android version history*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Android_version_history

[576] Wikipedia. *Clean room design*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Clean_room_design

[577] Wikipedia. *Community Memory*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Community_Memory

[578] Wikipedia. *Comparison of Android devices*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Comparison_of_Android_devices

[579] Wikipedia. *Comparison of Symbian devices*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/List_of_Symbian_devices

[580] Wikipedia. *Differential GPS (DGPS)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Differential_GPS

[581] Wikipedia. *e-Government*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/EGovernment

[582] Wikipedia. *Fast Fourier transform (FFT)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Fast_Fourier_transform

[583] Wikipedia. *Geocoding*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Geocoding

[584] Wikipedia. *GPL linking exception: The classpath exception*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/GPL_linking_exception#The_classpath_exception

[585] Wikipedia. *Great-circle distance*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Great-circle_distance

[586] Wikipedia. *History of the iPhone*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/History_of_the_iPhone

[587] Wikipedia. *HTC Hero*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/HTC_Hero

[588] Wikipedia. *Linear pulse-code modulation (LPCM)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Linear_pulse_code_modulation

[589] Wikipedia. *Pink noise*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Pink_noise

[590] Wikipedia. *Reliability of Wikipedia*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Reliability_of_Wikipedia

[591] Wikipedia. *S60 (software platform)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/S60_%28software_platform%29

[592] Wikipedia. *Symbian*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Symbian

[593] Wikipedia. *Symbian Ltd.* Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Symbian_Ltd.

[594] Wikipedia. *Think globally, act locally*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/Think_globally,_act_locally

[595] Wikipedia. *Waveform Audio File Format (WAVE or WAV)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/WAV

[596] Wikipedia. *White noise*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/White_noise

[597] Wikipedia. *World Geodetic System (WGS)*. Wikimedia Foundation.
URL: http://en.wikipedia.org/wiki/World_Geodetic_System

[598] Wikipedia contributors. *Wikipedia. The Free Encyclopaedia*. Wikimedia Foundation.
URL: http://wikipedia.org

[599] W. Williams, E. Beach, and M. Gilliver. "Clubbing: The cumulative effect of noise exposure from attendance at dance clubs and night clubs on whole-of-life noise exposure". In: *Noise and Health* 12.48 (2010), pp. 155–158.
ISSN: 1463-1741
DOI: 10.4103/1463-1741.64970

[600] Windows to the Universe. *Great World Wide Star Count*. National Earth Science Teachers Association (NESTA).
URL: http://www.windows2universe.org/citizen_science/starcount

[601] Allison Woodruff and Jennifer Mankoff. "Environmental Sustainability". Guest Editors' Introduction. In: *IEEE Pervasive Computing* 8.1 (Jan. 2009), pp. 18–21.
ISSN: 1536-1268
DOI: 10.1109/MPRV.2009.6

[602] World Commission on Environment and Development. *Our Common Future*. United Nations. Also known as the *Brundtland Report*. Oxford Paperbacks. Oxford University Press, Apr. 1987. 416 pp.
ISBN: 978-0-19-282080-8
URL: http://ukcatalogue.oup.com/product/9780192820808.do

[603] World Health Organization (WHO). *Guidelines for Community Noise*. Ed. by Birgitta Berglund, Thomas Lindvall, and Dietrich H. Schwela. 1999.
URL: http://www.who.int/docstore/peh/noise/guidelines2.html

[604] *World Wide Fund for Nature (WWF)*.
URL: http://wwf.org

[605] World Wide Web Consortium (W3C). *HTML5: A vocabulary and associated APIs for HTML and XHTML – W3C Working Draft*.
URL: http://www.w3.org/TR/html5

[606] Xamarin. *Mono for Android*. Initially marketed by Novell, Inc. as *MonoDroid*.
URL: http://android.xamarin.com

[607] Takahiro Yamamoto, Shunsuke Saruwatari, and Hiroyuki Morikawa. "Energy-Efficient Upload Engine for Participatory Sensing". In: *Proceedings of IWUSP 2010, the International Workshop on Ubiquitous Service Platforms (Kyoto, Japan, October 5, 2010)*.
URL: http://www.mlab.t.u-tokyo.ac.jp/publications/2010/#international_conf

[608] Tingxin Yan, Vikas Kumar, and Deepak Ganesan. "CrowdSearch: Exploiting Crowds for Accurate Real-time Image Search on Mobile Phones". In: *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services (San Francisco, CA, USA, June 15-18, 2010)*. New York, NY, USA: ACM, pp. 77–90.
ISBN: 978-1-60558-985-5
DOI: 10.1145/1814433.1814443

[609] Ed Yong. "City songbirds change their tune". In: *The New Scientist* 197.2649 (Mar. 29, 2008), pp. 33–35.
ISSN: 0262-4079
DOI: 10.1016/S0262-4079(08)60792-7

[610] Yucat Mobile Business Solutions. *BuitenBeter*.
URL: http://www.buitenbeter.nl

[611] Mei Zhang and Jian Kang. "Towards the evaluation, description, and creation of soundscapes in urban open spaces". In: *Environment and Planning B-Planning & Design* 34.1 (Jan. 2007), pp. 68–86.
DOI: 10.1068/b31162

[612] Aymeric Zils and Francois Pachet. "Automatic Extraction of Music Descriptors from Acoustic Signals Using EDS". In: *Presented at the 116th Audio Engineering Society Convention (May 8-11, 2004, Berlin, Germany)*.
URL: http://www.aes.org/e-lib/browse.cfm?elib=12768