Strong Code Mobility

KATHOLIEKE UNIVERSITEIT LEUVEN

VRIJE UNIVERSITEIT BRUSSEL • SCIENTIA VINCERE TENEBRAS •

LIMBURGS UNIVERSITAIR CENTRUM
IN HET CENTRUM VAN DE KENNIS

UNIVERSITEIT GENT

Leuven, 14 oktober 2004

Why Code Mobility?

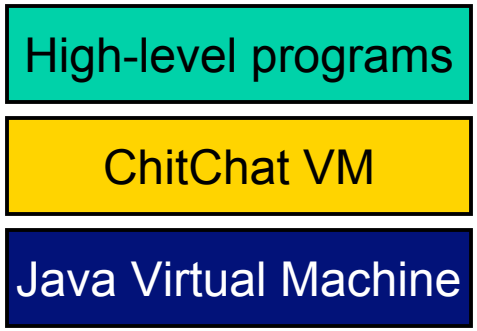
- Relocation of services is necessary in environments where the context frequently changes
- Users moving about geographically
- Collaborating service components need to migrate independently
- Migration must be seamless

Work Package on Code Mobility

- Strong Mobility
- Progressive Mobility
- Smart Mobility

Proof-of-concept high-level virtual machine

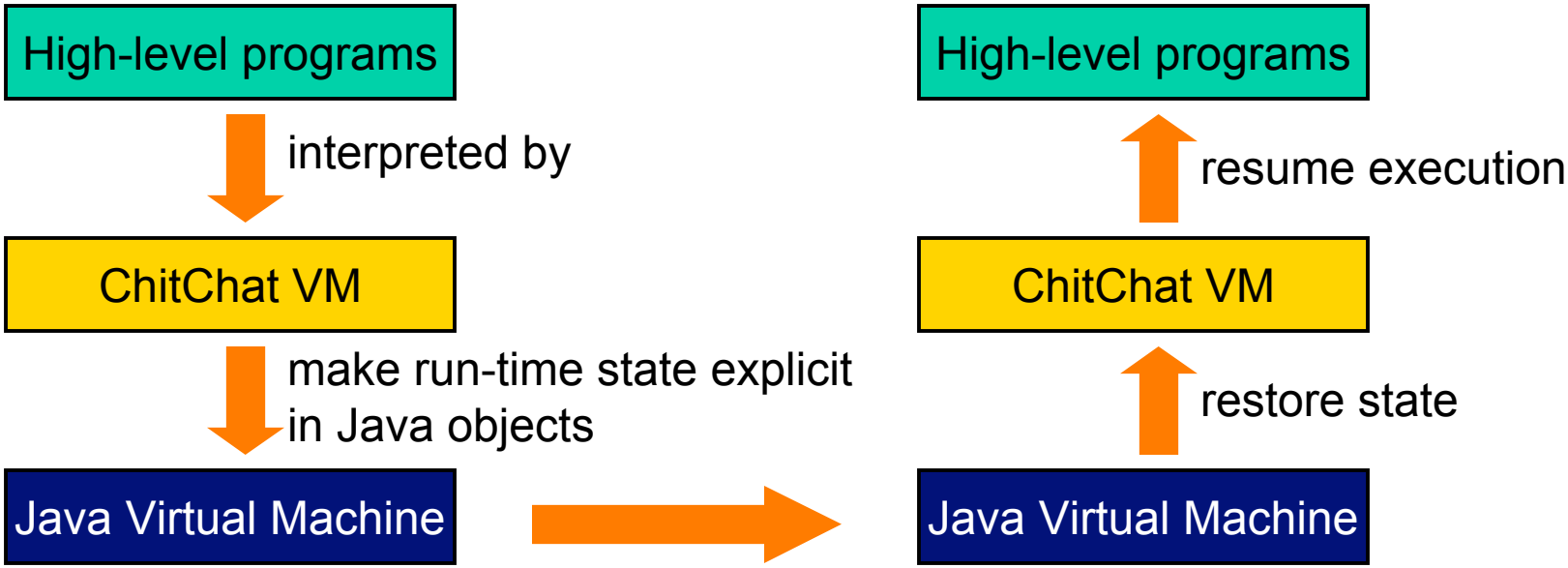
supporting strong code mobility



Strong Mobility: Approach

Proof-of-concept high-level virtual machine

supporting strong code mobility



serialize state to bytestream
send stream through Java I/O sockets

Types of Mobility

	Data Context	Control Context	Resources Context
Weak Mobility	⊘	⊘	⊘
Semi Strong Mobility	✓	⊘	⊘
Strong Mobility	✓	✓	⊘
Full Mobility	✓	✓	✓

Java Applets

Most Middleware Solutions

Threads ‘become’ java.io.Serializable

Process Migration (for e.g. load-balancing)

Why Strong Code Mobility?

Semi strong mobility
is far less expressive…

In an AmI environment, you
cannot anticipate every move!

should be executed
at the remote location…

```
public void m() {
    Object x = ...;
    o.n(this, x);
    // code hereafter never executed!
}
public void afterMove(Object x) {
    // perform some computation with x
}
```

```
public void n(Object obj,
              Object x) {
    // ...
    move(obj, someLocation);
    obj.afterMove(x);
}
```

Why not just use ‘Java’?

- No provisions for mobility
- Middleware/language extensions
  - interfere with standard Java semantics
  - often give up JVM compatibility
- Technical problems with recursive transmission of classes
- A Virtual Machine can abstract from the underlying host system

Move Considered Harmful

- Imagine combinations of…
  - regular control flow (if, while, …)
  - late binding polymorphism
  - meta-programming, reflection, aspects
  - move
- Which objects will be residing where?

Conjecture: move is the 'goto' of mobility

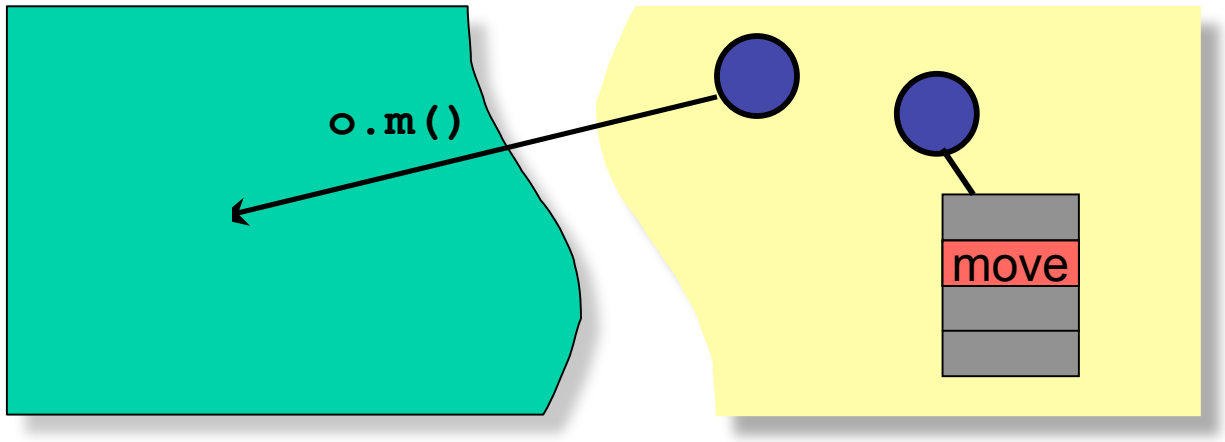
```
move(obj,134.184.43.120)  |  goto 0xff408a7e
move(doc,th
```

Wanted: Structured Mobility

We need abstractions to control the "loci of objects"

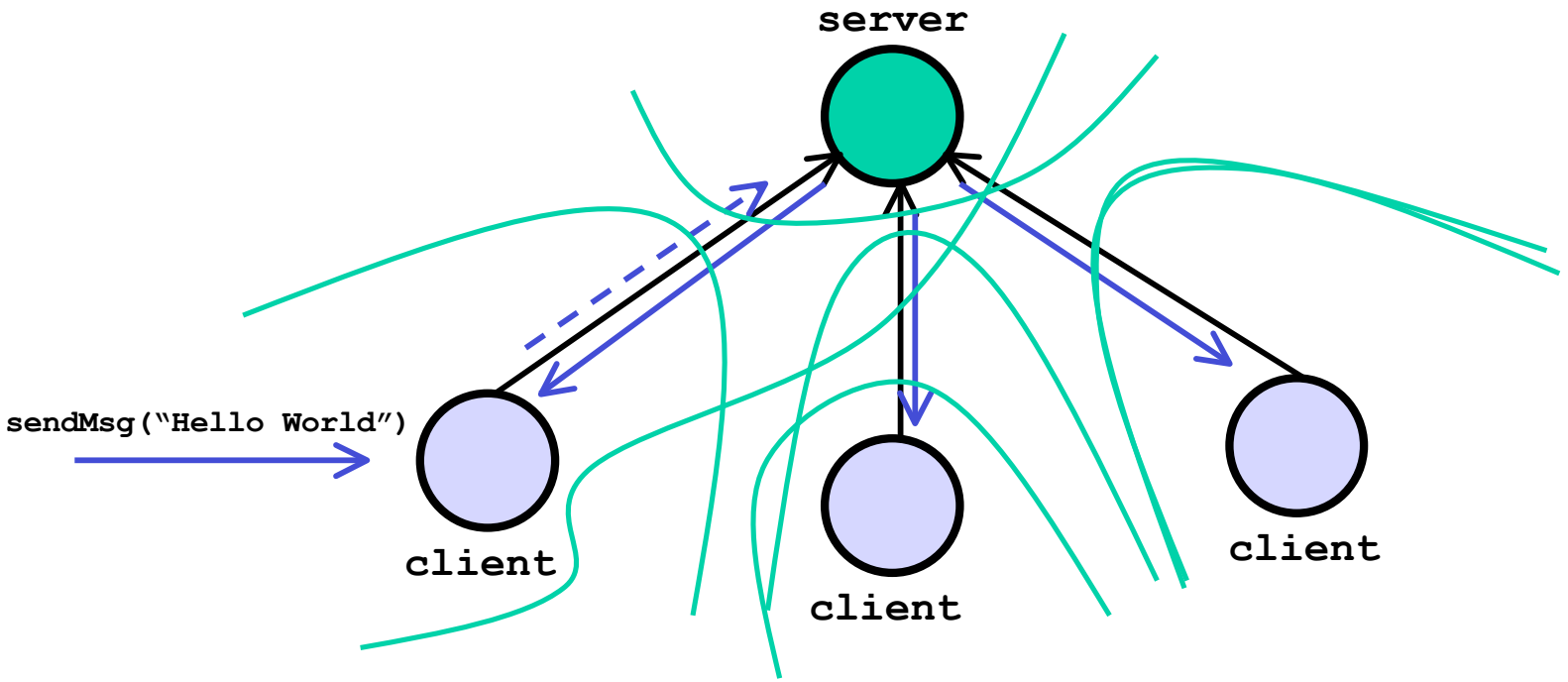
ChitChat: Structured Mobility

- Model based on active objects
- New kind of method ‘modifier’: **`move`**
- Move methods ‘pull’ objects from one VM to another:



Demo: Chat Client Application

- Simple client-server architecture
- Server automatically relocated to host of ‘most popular’ client



Demo: Chat Client Application

```
active object chatServer {
   Object[] clients;
   int occupancy, maxClients, max;

   public chatServer(channel, maxClients) {
      clients = new Object[maxClients];
      occupancy = max = 0;
      this.maxClients = maxClients;
      this.register(channel);
   }

   public move void come(nam) {
      System.out.println("arrived at client "+nam);
   }

   public Object registerClient(nam) {
      download new chatClient(nam);
   }
}
```

a chatServer is an active object…

…with a reference to its connected clients

move method used by clients to pull the server towards them

weak mobility: client code can be downloaded by a remote machine

Demo: Chat Client Application

```
active object chatClient extends chatServer {
    String nam; int count;

    public chatClient(nam) {
        this.nam = nam; count = 0;
        super {
          if (occupancy == maxClients)
              error("Sorry, channel is full");
          else
              clients[++occupancy] = this;
        }
    }

    public void receiveMsg(from, msg) {
        System.out
    }

    ...
}
```

clients have a dynamic relation with the server

```
public void sendMsg(msg) {
    if (++count > super.max) {
       super.come(nam);
       super.max = count;
    }
    super {
      for (int i=0, i < occupancy, i++)
       clients[i].receiveMsg(nam, msg);
    }
}
```

if I am the most active client, pull the server towards me

broadcast the message to all other clients