SOUL and Smalltalk
Just Married

Kris Gybels

Vrije Universiteit Brussel
DMP = Declarative Meta-Programming

SOUL = Smalltalk Open Unification Language
“Language symbiosis” : allowing transparent interactions between programs
Smalltalk term as a condition: to "prove", execute code

class(\(?x\)) if
var(\(?x\)),
\[ generate(\(?x\), [ System allClasses ] ) \]

class(\(?x\)) if
nonvar(\(?x\)),
\[ ?x isClass \]

subclass(\(?super\), \(?sub\)) if
class(\(?sub\)),
equals(\(?super\), \[ ?sub superclass \])

hierarchy(\(?root\), \(?child\)) if
subclass(\(?root\), \(?child\))
hierarchy(\(?root\), \(?child\)) if
subclass(\(?root\), \(?direct\)),
hierarchy(\(?direct\), \(?child\))

generate predicate: like member but for Smalltalk collections

Smalltalk term as value, executed during unification

Smalltalk objects "travel" to SOUL level
argumentArray := Array with: (Array with: #x with: Object).
evaluator := SOULEvaluator eval: 'if hierarchy(?x, ?y)' withArgs: argumentArray.
results := evaluator allResults.
ysolutions := results bindingsForVariableNamed: #y.
Order

price

\(^{\text{self totalPrice}} \times (100 - \text{self discount})/100\)

discount

(customer isLoyal) ifTrue: \(^{5}\).

(self nrItems > 20) ifTrue: \(^{10}\).

Customer

isLoyal

....

?order discount = 5 if
?order customer = ?customer &
?customer isLoyal

?order discount = 10 if
?order nrItems = ?nrItems &
?nrItems > 20

?customer isLoyal if
....
Changes to SOUL

• Syntactical: more like Smalltalk

  \begin{align*}
  \text{member(?item, ?list)} & \quad \Rightarrow \quad \text{?list contains: ?item} \\
  \text{member(?item, <?item | ?rest>)}. & \quad \Rightarrow \quad <?item | ?rest> contains: ?item. \\
  \text{member(?item, <?first | ?rest>) if} & \quad \Rightarrow \quad <?first | ?rest> contains: ?item if \\
  \text{member(?item, ?rest)} & \quad \Rightarrow \quad ?rest contains: ?item
  \end{align*}

• Syntactical: added = construct

  \begin{align*}
  \text{?order discount = 5 if} & \quad \Rightarrow \quad \text{?order customer = ?customer &} \\
  \text{?customer isLoyal}
  \end{align*}
?list containsSomethingRed if
?list contains: ?x &
?x isRed

containsSomethingRed(?list) if
member(?x, ?list)
[ ?x isRed ]

No rule for isRed, so send as message to object in ?x

Smalltalk term, send message isRed to object in ?x

Success/failure

Note: defining a rule for isRed would override message sending
Smalltalk->SOUL

Soul programmer would want to write
?p ancestor = ?a
or
?p hasAncestor: ?a

No method for ancestors so resolve as query

if ancestors = ?p

collection of results for parameter after = or FakeSingleItemCollection

query solutions
Useful concepts?

Language interaction versus language addition?

- Other experiences?
- General principles for MP possible? Art or science?