Orthographic Service Engineering
Contents

- Orthographic drawing
- Orthographic software modeling
- Multi-view manipulation
- Prototype tool
- Repository format
- Current status
A Simple "Modeling" Metaphor

- other engineering disciplines have a long and successful tradition of technical drawing
  - orthographic projection

- so why don't we do this in software engineering?
Orthographic Projection

- a long-established technique for depicting physical artifacts in a way which is
  - concise
  - precise
  - intuitive
  - permanent
  - modular
  - scaleable
  - composition-based
  - standardized

- a suitable metaphor for model driven development?
The Model versus the "Real Thing"

△ distinction between the model and the "real thing" is clear in other engineering disciplines
Traditional "Language" Technology

in software, the binary code is the closest we can get to the "real thing"
Orthographic Software Modeling

- mappings are driven by all three model
White Box vs. Black Box Views

- Black Box View
- White Box View
Bank Component Realization (KobrA)

**Structural Model**

Class Diagram:
- **Converter**
  - getRate
  - convertToEuro
  - convertFromEuro

Object Diagram:
- **Teller** accesses **Account**
  - createAccount
  - getAccount
  - closeAccount

**Activity Model**

Activity Diagram:
- **Teller**
  - getAccount
  - 

**Interaction Model**

Collaboration Diagrams:
- **Teller**
  - getAccount
  - convertToEuro
  - convertFromEuro
- **Converter**

**Activity Model**

Activity Diagram:
- **Teller**
  - getAccount
  - 

**Interaction Model**

Collaboration Diagrams:
- **Teller**
  - getAccount
  - convertToEuro
  - convertFromEuro
- **Converter**

**Activity Model**

Activity Diagram:
- **Teller**
  - getAccount
  - 

**Interaction Model**

Collaboration Diagrams:
- **Teller**
  - getAccount
  - convertToEuro
  - convertFromEuro
- **Converter**

**Activity Model**

Activity Diagram:
- **Teller**
  - getAccount
  - 

**Interaction Model**

Collaboration Diagrams:
- **Teller**
  - getAccount
  - convertToEuro
  - convertFromEuro
- **Converter**
White Box vs. Black Box Components

black box components

white box components
Orthographic Decomposition

views of composite

views of component
Teller Component Specification (KobrA)

**Structural Model**

```plaintext
<<subject>>
Teller

noOfAccounts: Integer := 0
createAccount
getAccount
closeAccount

PersistentAccount
  accountId: String
  balance: Float
  Denom: String

Class Diagram
```

**Functional Model**

**Operation Specifications**

<table>
<thead>
<tr>
<th>Name</th>
<th>createAccount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Informal Description</td>
<td>An account is opened in a particular currency for a customer with a particular name, and the Account ID is returned</td>
</tr>
<tr>
<td>Constraints</td>
<td></td>
</tr>
<tr>
<td>Receives</td>
<td>name: String, currency: String</td>
</tr>
<tr>
<td>Returns</td>
<td>A String with the ID of the account</td>
</tr>
<tr>
<td>Changes</td>
<td>teller</td>
</tr>
<tr>
<td>Assumption</td>
<td>There is an exchange rate for the specified currency</td>
</tr>
<tr>
<td>Result</td>
<td>A new account with a unique ID in the denomination, currency, has been generated</td>
</tr>
</tbody>
</table>

The name of the customer has been stored in account
The account ID has been returned

**Behavioural Model**

```plaintext
Empty
createAccount

HasAccounts
  noOfAccounts: Integer := 0
  createAccount/
  getAccount/
  closeAccount [noOfAccounts = 1]

Statechart Diagram
```
Abstraction

abstract view

concrete (detailed) view
Aspect-Oriented Programming

"orange" aspect elaboration

"blue" aspect elaboration
Projection Example

Surface 1

Mag: 0  X-Ray: 0  Filter: None  Projection: front  Render: UML  Cpts: 0
Projection Example: All Views Displayed

<table>
<thead>
<tr>
<th>Mag: 0</th>
<th>X-Ray: 0</th>
<th>Filter: None</th>
<th>Render: UML</th>
<th>Cpts: 0</th>
</tr>
</thead>
</table>

- **Projection: Front**
- **Projection: Side**
- **Projection: Variability**
- **Projection: Behavioural**
Bank Example

**Bank**

**Mag: 0**  
**X-Ray: 0**  
**Filter: None**  
**Render: UML**  
**Cpts: 0**

**Operation Specifications**

### Decision Table

<table>
<thead>
<tr>
<th>ID</th>
<th>Question</th>
<th>Subject</th>
<th>Resolution</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Is a customer allowed to overdraw their account up to a certain limit?</td>
<td>Overdraw Limit</td>
<td>(not defined)</td>
<td>Resolve structural diagram: withdrawal decision 1: yes Resolve operation spec: withdrawal decision 1: yes</td>
</tr>
<tr>
<td>2</td>
<td>Is it an international bank that handles different currencies?</td>
<td>Currencies</td>
<td>(not defined)</td>
<td>Resolve structural diagram: withdrawal decision 1: no Resolve operation spec: withdrawal decision 1: no</td>
</tr>
</tbody>
</table>

**Projection: Functional**

**Projection: Structural**

**Projection: Variability**

**Projection: Behavioural**
Bank Example: Structural View

Bank

noOfAccounts : Integer := 0

createAccount()
deposit()
viewAccount()
withdraw()
closeAccount()
setRate()
convertToEuro()
convertFromEuro()

Account

accountID : String
ownerName : String
balance : Float
denom : String
Limit : String

1 manages *
Component Modeling Tool Support

- AFD
- UML
- test cases
- OpSpec
- RegEx
- code
- software component
- UML classes
- Java Source
- Behavior
Ideal World
Pragmatic Solution

UML classes

Behavior

test cases

code

SMM

test cases code

UML classes

Behavior

Java source

22
Component Oriented Navigation

Component
- Bank
- Converter
- Teller

Abstraction
- Specification
- Realization
- Implementation

Projection
- Structural
- Functional
- Behavioural
- QoS

Cell
Operation Specification Editor

withdraw

**General**

Name: withdraw

Description: An amount of money in a particular currency is withdrawn from an account.

Parameters: accountID: Integer

amount: Real

currency: String

Return: Boolean

**Edit Constraint**

Name: Account-Exists-Precondition

Constraint: account->exists(accountID=id)

Description: Make sure that account exists

OCL is valid
OMG Four Layer Model

M3 (MOF)

Class

«instanceOf» «instanceOf» «instanceOf»

M2 (UML)

Attribute  Class  Instance

«instanceOf»  «instanceOf»  «instanceOf»  «instanceOf»

M1 (User model)

Video
+title: String

Video
: Video
title = "2001: A Space Odyssey"

«instanceOf» «snapshot»

M0 (Run-time instances)

aVideo
Orthographic Classification Architecture

The MOF

The UML
- rules
- "knowledge" patterns

Language Definition

Model

System or world

[Atkinson and Kuehne, 1996 …]
Java OCL (JOCL)

A professor should only teach students at his university

OCL

inv: self.course.student.university->asSet().including(self.university)->asSet()->size() = 1

JOCL

self.navigate("course").navigate("student").navigate("university").asSet().including(self.navigate("university")).asSet().size() == 1
Repository Representation Format
AST and Surface Syntax Mismatch

Model as instances

Model as types

Groningen

Smith

University

Professor

Association

Association End

2

employs

employs
Uniform Syntax

OCL frame constraints

Class <-> Connection

University -- employs --> Professor

Groningen -- employs --> Prof. Smith
## Current Editors and Visualizers

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>Composition, Abstraction, Projection, Version, Product Line, Metalevel, …</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Projection</th>
<th>Structural</th>
<th>Functional</th>
<th>Behavioral</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstraction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specification</td>
<td>• UML class diagram • taxonomy • component description</td>
<td>• operation specification • activity template</td>
<td>• UML state chart • regular expression</td>
<td>…</td>
<td></td>
</tr>
<tr>
<td>Realization</td>
<td>• UML class diagram</td>
<td>• UML interaction diagram</td>
<td>• UML activity diagram</td>
<td>…</td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td>• source code view</td>
<td>-</td>
<td>-</td>
<td>…</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
IDE Status (Aristaflow Project)

- **Completed**
  - basic framework
  - calls editors
  - Component CRUD

- **In Progress**
  - persistence (bug fixes)
  - GUI improvements to ease navigation

- **To Do**
  - create dimensions, create perspectives dynamically
  - documentation
  - distributable packaging
  - synthesizers and checkers
Further Information

Web Pages
- swt.informatik.uni-mannheim.de
- www.merobase.com

Contact
- atkinson@informatik.uni-mannheim.de