Model Driven Ecological Interface Creation: The Constraints Model

Alexandre Moïse
Jean-Marc Robert
Department of Mathematics and Industrial Engineering
École Polytechnique de Montréal
What is an ecological interface?

- User interface
  - Ecological psychology
  - Fault detection and diagnosis
  - Structure of an environment
    - Functional hierarchy
    - Constraints
  - Emphasis on relations
  - Direct manipulation
Traditional interface

Pawlak, W. S., Vicente, K. J. "Inducing effective operator control through ecological interface design", *Int. J. Human-Computer Studies*, vol. 44, pp. 653-688, 1996.
Ecological interface

Pawlak, W. S., Vicente, K. J. "Inducing effective operator control through ecological interface design", *Int. J. Human-Computer Studies*, vol. 44, pp. 653-688, 1996.
The design (and implementation) gap

Analysis model

Source code

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class uneServlet extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
            throws ServletException, IOException {
        // Ajouter code
    }

    public void doPost(HttpServletRequest request,
                        HttpServletResponse response)
            throws ServletException, IOException {
        // Ajouter code
    }
}
```
Abstraction-decomposition hierarchy
User interface modeling

- Abstract presentation
- Concrete presentation
- Final presentation
User interface modeling

CIM

PIM

PSM

Presentation model
  - Abstract presentation model
  - Concrete presentation model

Logic model

Data model

Process model

Domain model

Presentation code

Logic code

Data code
Modeling the abstract presentation

- Abstract individual component
  - Concrete form: text area, drop-down list, check box

- Abstract container
  - Concrete form: section, grouping, frame
Abstract syntax
Abstract syntax
Abstract syntax
Abstract syntax
Abstract syntax

```
leftOperand != rightOperand
```

Diagram:
- **Arithmetic** (apm.operator.arithmetic)
  - **Continuous** (apm.variable)
    - **Addition** (apm.operator.arithmetic)
    - **Subtraction** (apm.operator.arithmetic)
    - **Multiplication** (apm.operator.arithmetic)
    - **Division** (apm.operator.arithmetic)
Abstract syntax

Diagram:

- `leftOperand != rightOperand`
- `Continuous (apm.variable)`
- `Logical (apm.operator.logical)`
- `RightSide (apm.constraint)`
- `LowerThan (apm.operator.logical)`
- `Equals (apm.operator.logical)`
- `GreaterThan (apm.operator.logical)`
Abstract syntax
Concrete syntax

- **Concept**

- **Operation**

- **Variable**
Application (structure and constraints)
Application (functional hierarchy)

Client.targetRateOfReturn < Portfolio.rateOfReturn
Client.riskTolerance > Portfolio.risk

Portfolio.risk = SUM OwnedFund.weightedRisk
Portfolio.rateOfReturn = SUM OwnedFund.weightedRateOfReturn

OwnedFund.weightedRisk = Fund.risk \times OwnedFund.weight
OwnedFund.weightedRateOfReturn = Fund.rateOfReturn \times OwnedFund.weight

OwnedFund.weight = OwnedFund.value \div Portfolio.value

Portfolio.value = SUM OwnedFund.value
OwnedFund.value = Fund.currentPrice \times OwnedFund.nbShares
Conclusion

- Another one of the few applications of DSML to UI modeling

- Better understanding of ill-defined concepts
  - Constraint, Functional hierarchy

- Future work
  - Concrete presentation, Final presentation
  - Transformations