Using Obfuscating Transformations for Supporting the Sharing and Analysis of Conceptual Models

Dr. Hans-Georg Fill

Research Group Knowledge Engineering

University of Vienna

OPEN | MODEL Initiative

FWF

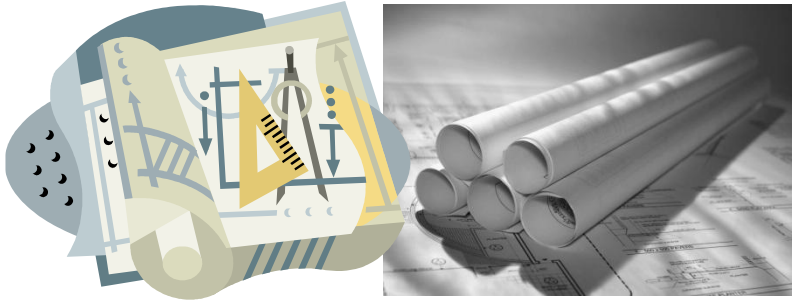
universität wien

Agenda

- Motivation
- Foundations
- Obfuscating Transformations for Conceptual Models
- Use Case
- Conclusion and Outlook

Motivation

- Several initiatives for sharing of conceptual models, reference models, modeling languages and modeling tools
- Based on principles similar to Open Source
- Goals:
  - Provide large repositories of models for learning, research, and development purposes
  - Establish a community of practitioners and academics
  - Exchange of knowledge and expertise
  - Access to IT concepts and tools



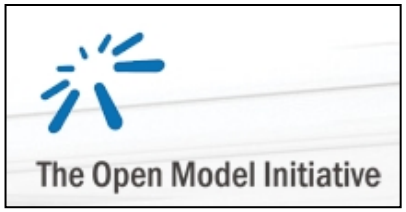
Example: Open Model Initative

"…intends to establish a community of people who focus on the creation, maintenance, modification, distribution, and analysis of models.“

(Karagiannis et al., 2008, p.3)

Current stats:

- approx. 17 categories of reference models available on www.openmodels.org that include >40 model instances
- approx. 22 projects that develop/provide modeling tools and services on www.openmodels.at



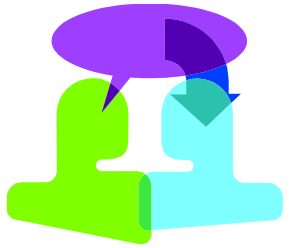
www.openmodels.org

www.openmodels.at

Aspects of Sharing Models

- Intellectual property aspects of models
- Confidential information contained in models:
  - Work practices
  - Business relationships
  - Other information that is potentially harmful when disclosed to competitors
- Benefits of sharing models:
  - Enable comparisons, benchmarks
  - Receive feedback for improvements
  - Provide documentation for interaction with customers and suppliers

Balance?



Foundations

Privacy Preserving Data Mining (PPDM)

- Sanitizing data sets for shielding sensitive information and disallowing derivation of certain associations
  - Before the mining process
  - After the application of data mining techniques
- Preservation of properties similar to original data sets to receive reasonably accurate results

- Secure multi-party computations:
  - Several parties provide input for computations, e.g. benchmarking
  - Input of parties is not disclosed

Source Code Obfuscation

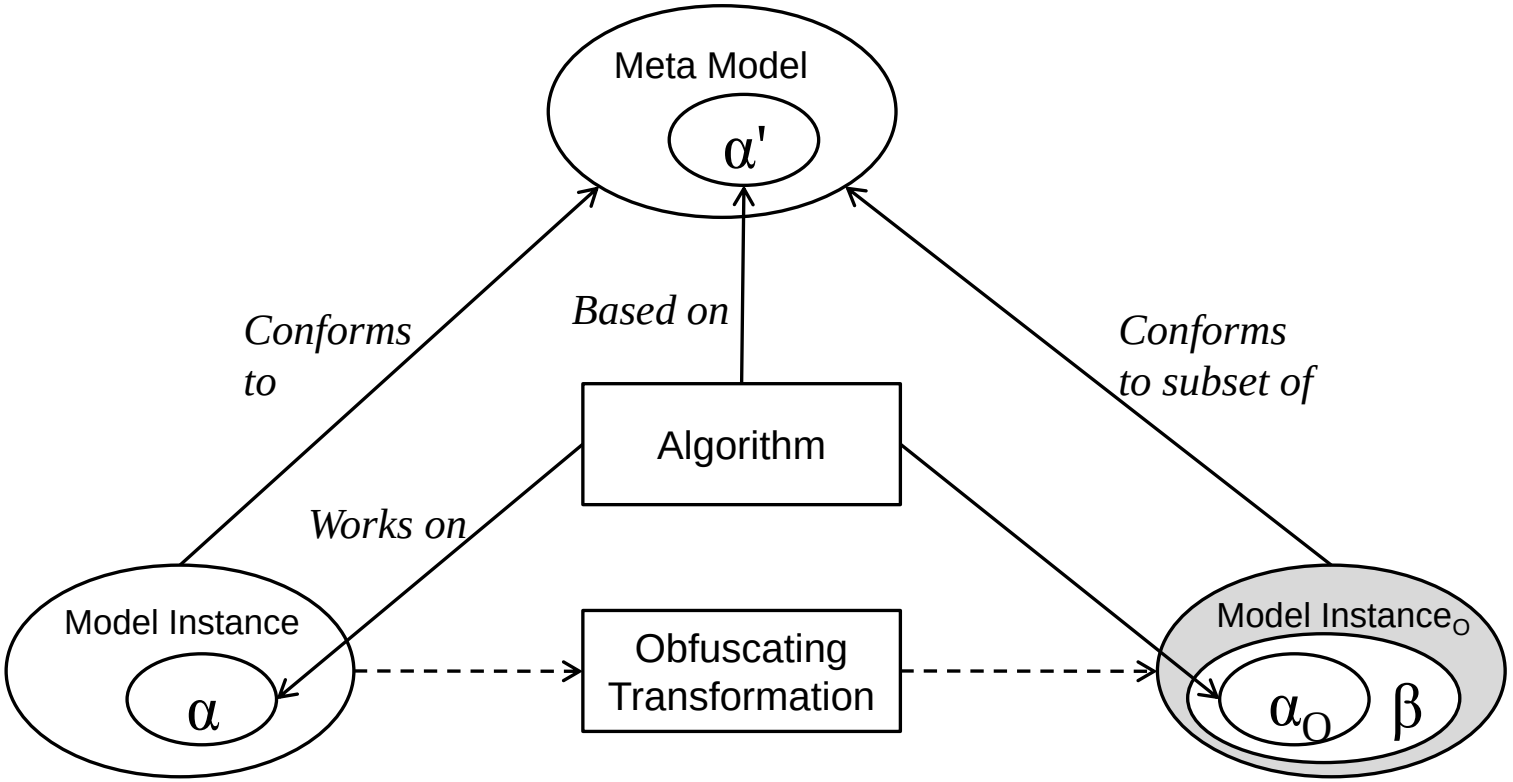
- Convert source code of programs into one with same observable behavior:
  - Same input produces the same output
  - Input is harder to understand and reverse-engineer
- Three types of obfuscations:
  - Layout transformations: change the formatting of source code and identifiers of variables
  - Control transformations: modify the control flow of a program to hide its functioning
  - Data transformations: target the data structures of an application

Obufscating Transformations for Conceptual Models

Goals of Obfuscating Transformations for Conceptual Models

- Hide confidential and potentially harmful information in models
- Adapt techniques from PPDM and source code obfuscation
- Preserve main structure and contained semantic information as much as possible
- Preserve applicability of certain algorithms
- Focus on one-way and highly resilient transformations
- Avoid adding complexity to models to maintain comprehensibility

Concept for Applying Obfuscating Transformations

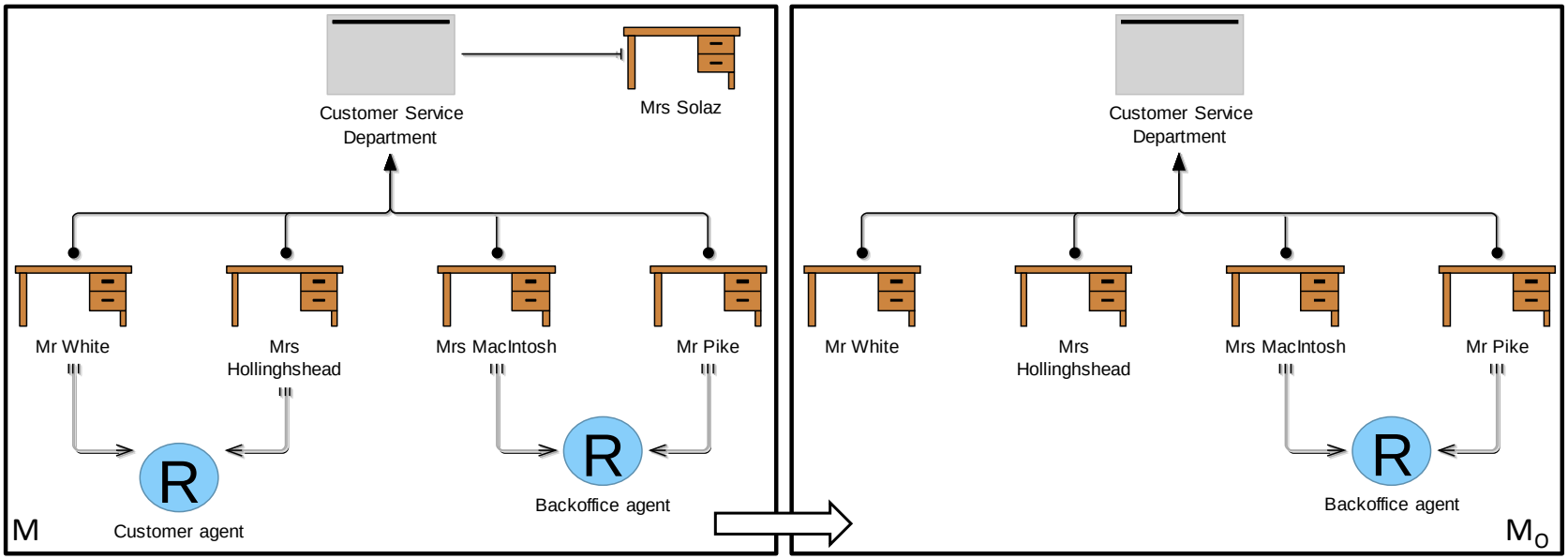


Types of Transformations

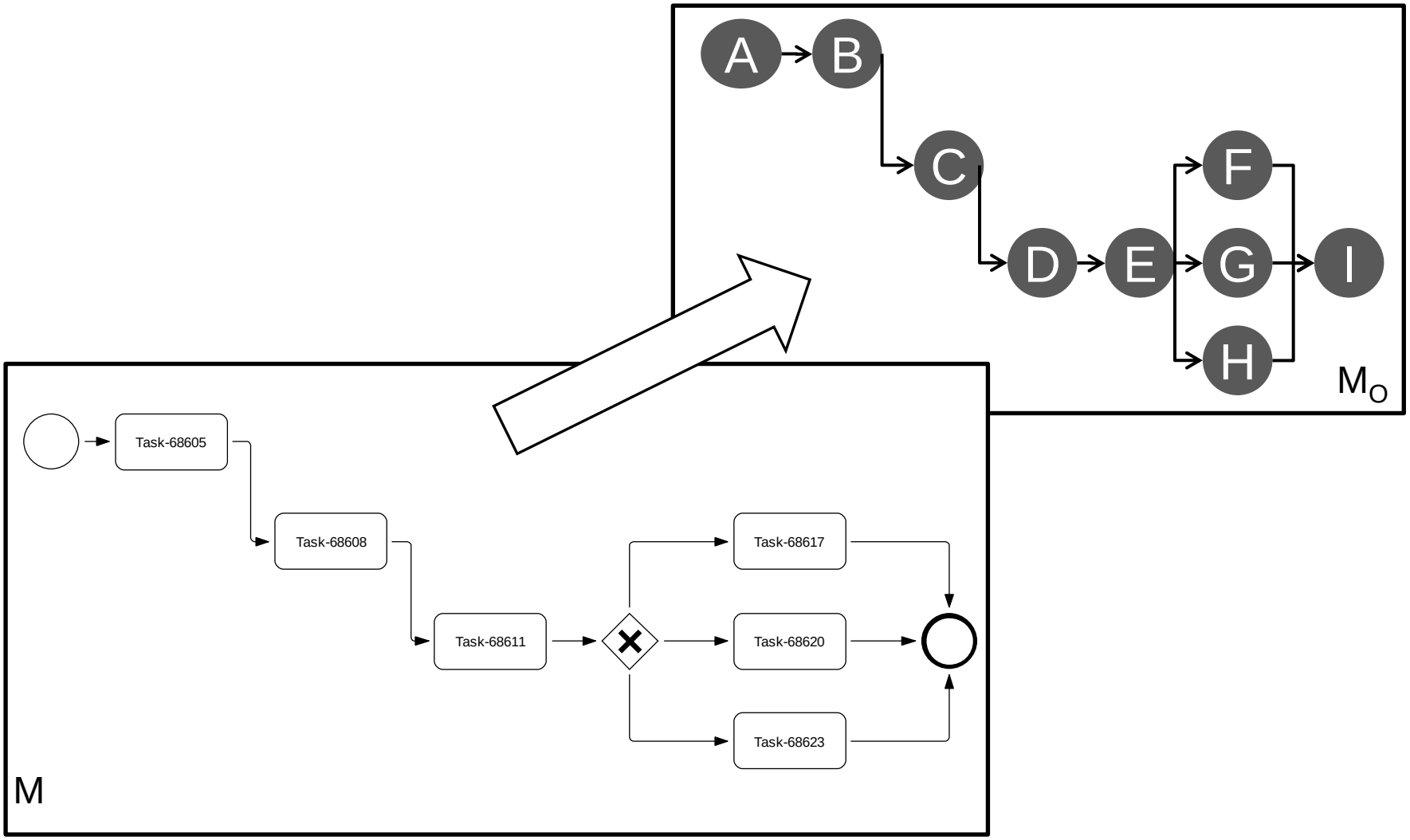
- Representation transformations
- Structural transformations
- Data transformations
- Semantic obfuscation transformations

Representation Transformations I

- Derived from layout transformations in source code obfuscation:
  - Hiding information
  - Abstraction of models

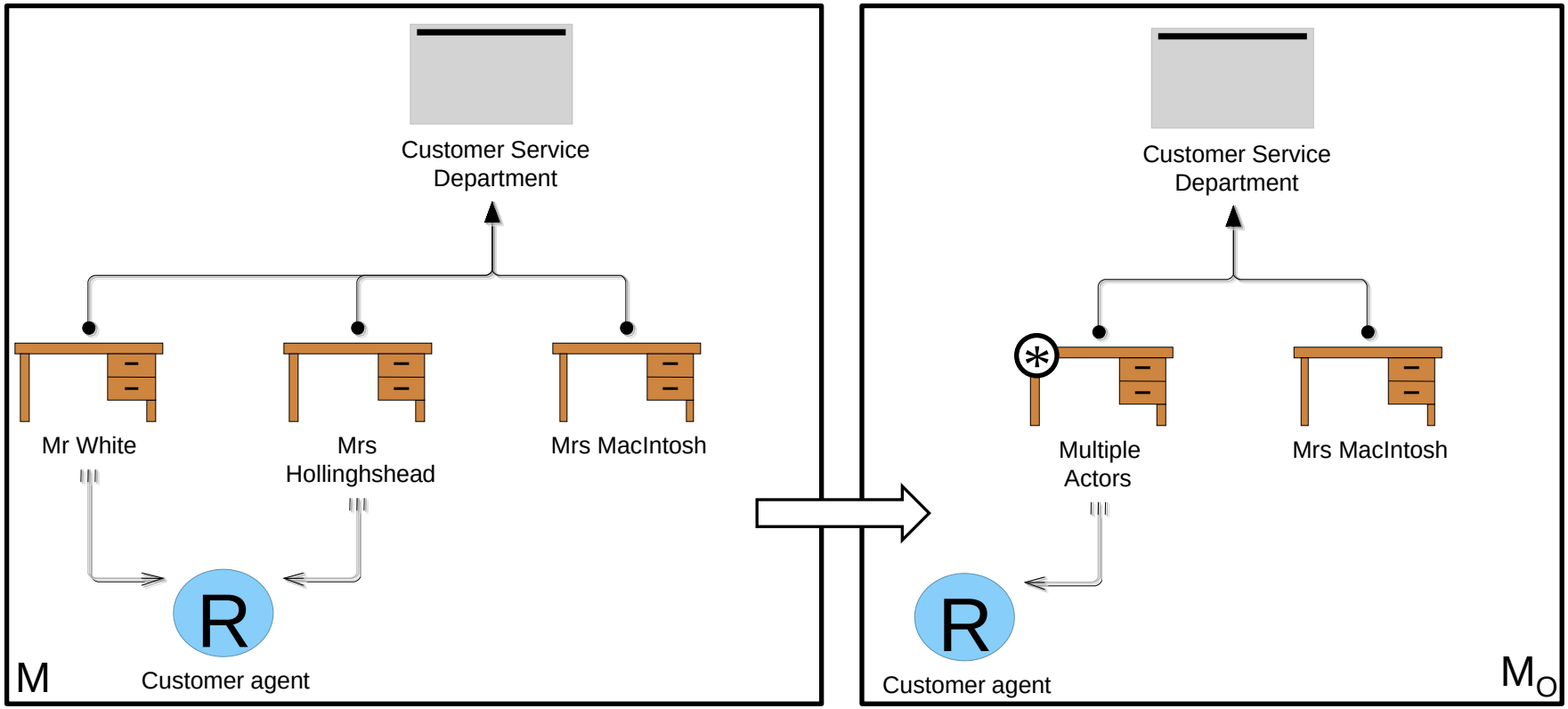


Representation Transformations II

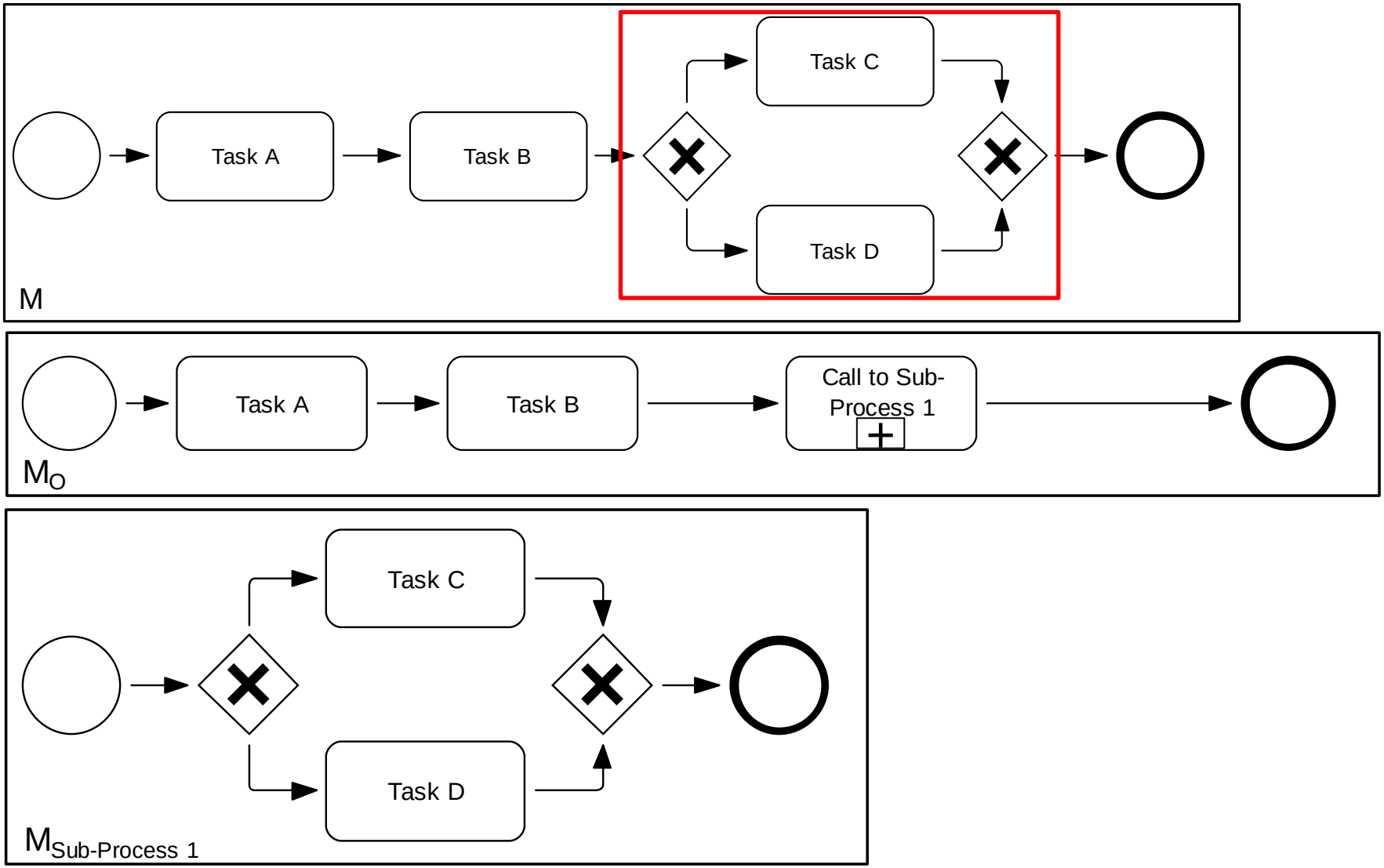


Structural Transformations I

- Change the structure and complexity of the models to hide model details
- Use nesting mechanisms to embed models in each other and hide parts of models

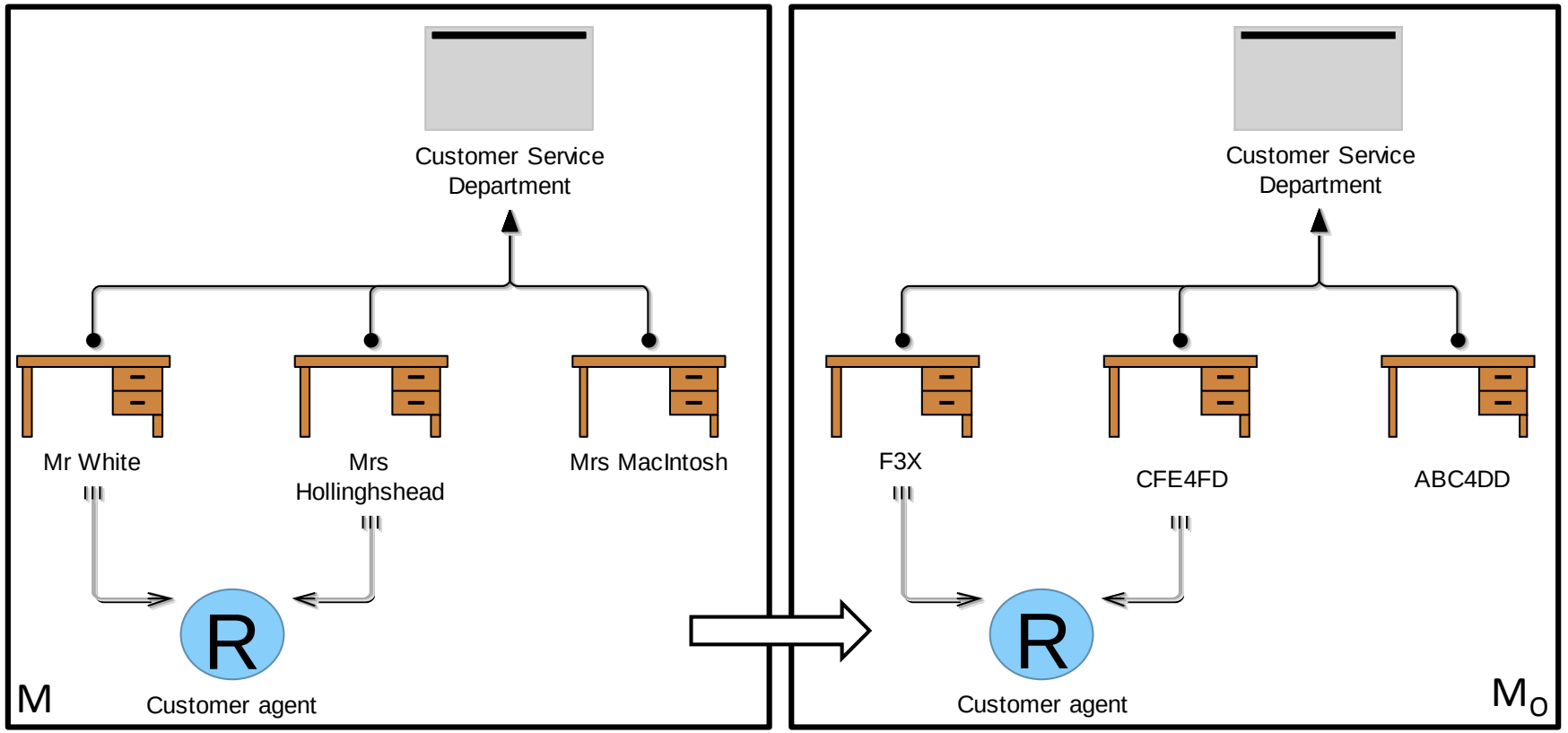


Structural Transformations II

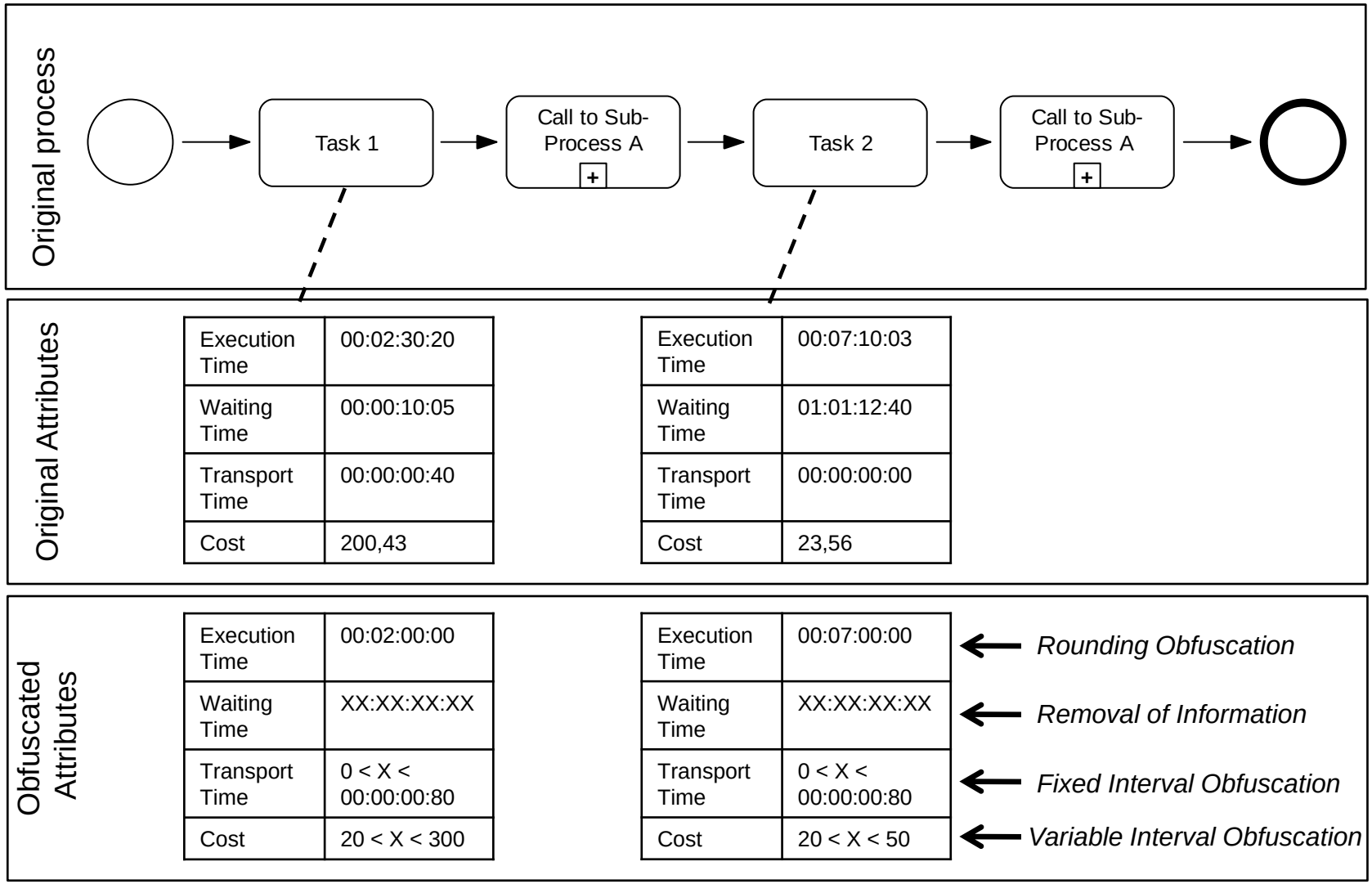


Data Transformations I

- Obfuscation of labels and attributes of elements:
  - Scrambling
  - Numerical Value Transformations

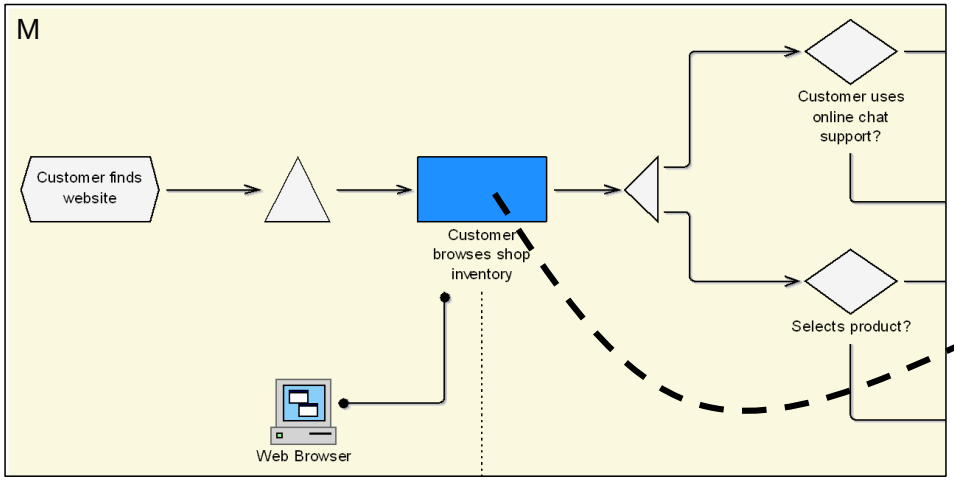


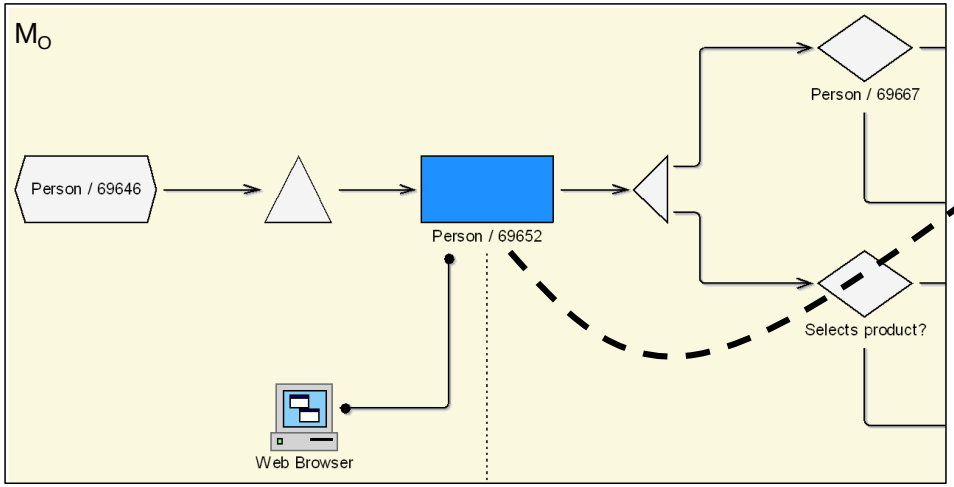
Data Transformations II



Semantic Obfuscation Transformation

- Using the subsumption hierarchy of an ontology and semantic annotations





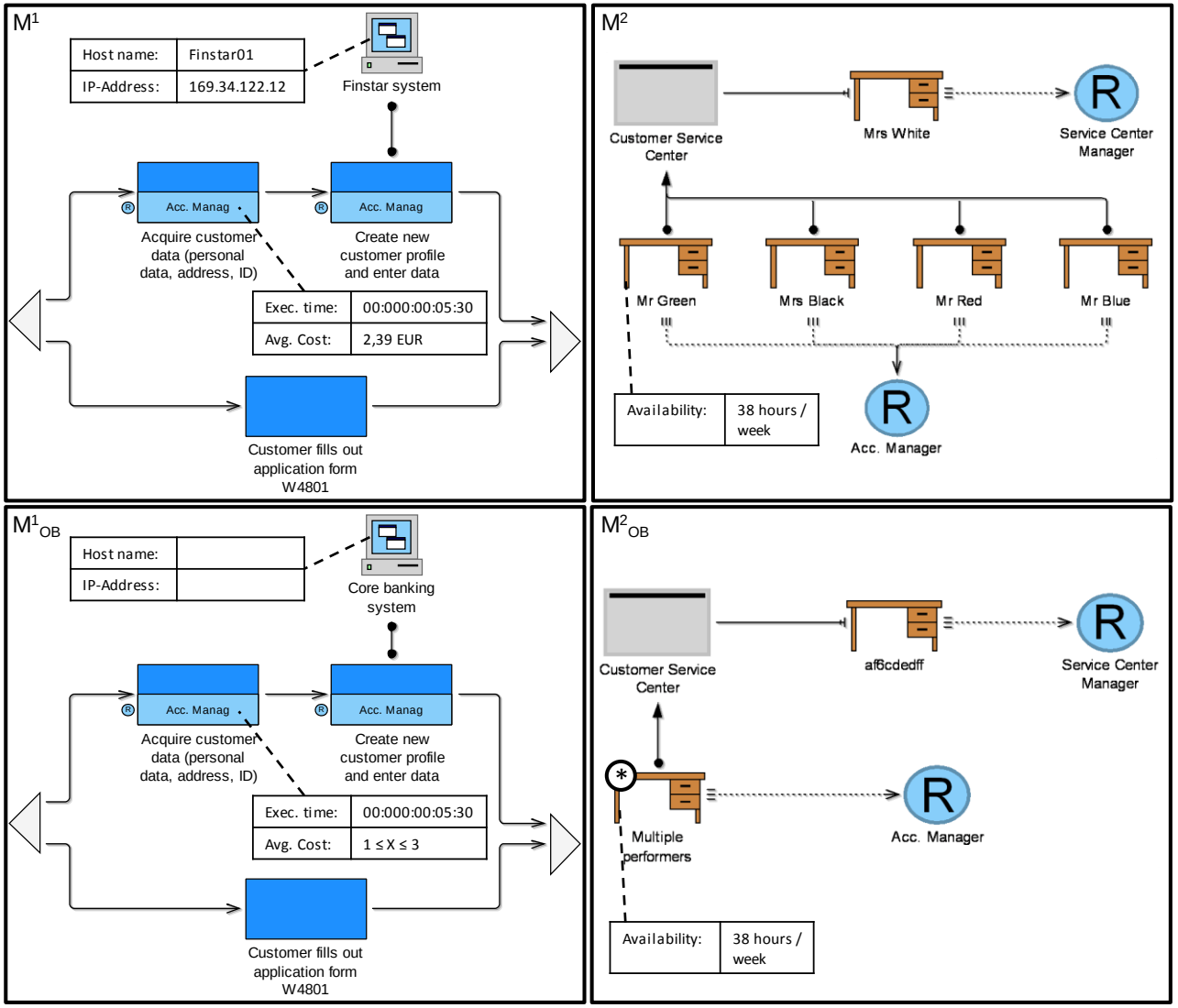
Ontology in OWL

```
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:    <http://www.w3.org/2002/07/owl#> .

default:Customer
    a       owl:Class ;
    rdfs:subClassOf default:Person ;
    owl:disjointWith default:Employee .

<http://www.owl-ontologies.com/ObfuscationOntology.owl>
    a       owl:Ontology .

default:Employee
    a       owl:Class ;
    rdfs:subClassOf default:Person ;
    owl:disjointWith default:Customer .

default:Person
    a       owl:Class .
```

Application to a Use Case

- Based on a publicly available description of an account opening process of a Swiss bank
- Concrete issues that may necessitate obfuscations:
  - Competitive issues, i.e. information that should not be disclosed to competitors
  - Security issues, i.e. information that potentially permits to access non-public systems or could facilitate such access
  - Privacy issues, i.e. information that concerns personal data
- Application of the following techniques:
  - Hiding of information
  - Condensing of elements
  - Label scrambling
  - Variable interval obfuscation for process activity costs
  - Semantic obfuscation transformation for names of IT systems

Example Models from the Use Case



Conclusion and Outlook

- Obfuscating transformations allow to preserve the core structures of the models
- Algorithmic analyses of certain models parts are still feasible – e.g. in regard to the execution time and capacity analyses in the use case
- Also obfuscated models can provide a lot of value for research and knowledge exchange

Next steps:

- Detail transformations using mathematical notation
- Implement them in a modeling tool
- Apply them to further use cases for evaluation

Thank you for your attention!
hans-georg.fill@dke.univie.ac.at

OPEN | MODEL
Initiative

www.openmodels.org
www.openmodels.at

Tutorial „Metamodellierungsplattformen im Einsatz am Beispiel ADOxx® & SOM“ Modellierung 2012, 14.3.2012, Bamberg