Construction of Complex UML Profiles

UPM

ETSI Telecomunicación
Ciudad Universitaria s/n
Madrid 28040, Spain
mmiguel@dit.upm.es

dit UPM

Context of this work

- The present courseware has been elaborated in the context of the MODELWARE European IST FP6 project (http://www.modelware-ist.org/).
- Co-funded by the European Commission, the MODELWARE project involves 19 partners from 8 European countries. MODELWARE aims to improve software productivity by capitalizing on techniques known as Model-Driven Development (MDD).
- To achieve the goal of large-scale adoption of these MDD techniques, MODELWARE promotes the idea of a collaborative development of courseware dedicated to this domain.
- The MDD courseware provided here with the status of open source software is produced under the EPL 1.0 license.

UML Profiles : Table of content

- UML Extension of Complex Concepts
- Conceptual Models
- Metamodels Construction
- Profile Construction
- Mappings of Profiles and Metamodels

UML: a Language with Semantic Non-Strict (1)

- UML is a **general modeling language** that can be applied in very different domains, development phases, or technologies
  - Domain extensions (e.g. UML for the description of Avionic software architectures)
  - Methodological adaptations (e.g. integration of incremental life cycles in modeling driven development)
  - Technical extensions (e.g. modeling real-time applications)
  - Technological extensions (e.g. web server applications construction)
- **UML Family of Languages**
- Each adaptation requires **extend and restrict** the UML semantic to the specific domain
- Most of them are complex concepts that require complex notations for their description

UML: a Language with Semantic Non-Strict (2)

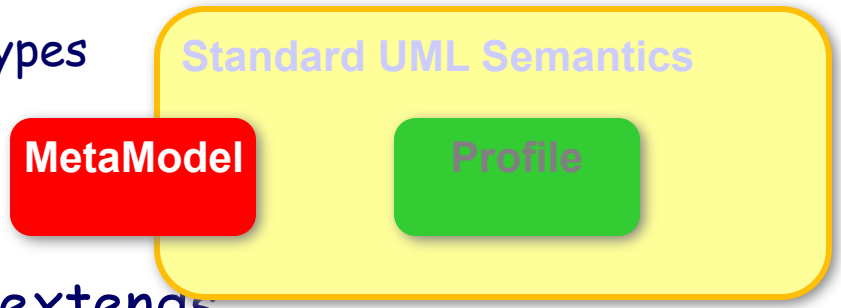
- The integration of new concepts in UML requires:
  - Design new notations for the description of new concepts
  - Integrate the new concepts with UML modeling notation
    - Reuse UML modeling elements for the description of new concepts
- In the construction of extension we must:
  - Identify the new concepts to be modeled
  - Design new notations for the description of new concepts
- UML Extension is a solution for the support of MDA in specific platforms, domains and techniques.

Environments to put in practice MDA

- Modeling languages is the central topic of MDA. Two approaches to support modeling languages:
  - **Languages for Modeling:** Meta-Modeling Frameworks for the description of modeling notations
    - Support of repositories, models interchange and management
  - **Profiling**: Profile Builders for the construction of Profiles models
    - Description of Stereotypes and attributes
    - Relations of stereotypes:
      - extension of metaclasses,
      - stereotype inheritances,
      - associations to metaclasses and stereotypes

UML Profile vs. UML Meta Models:
Two Different Approaches to Extend UML

- Profile: package that contains model elements that have been customized for a specific domain using
  - stereotypes
  - definition of attributes of stereotypes
  - constraints
  - model libraries
  - metamodel subset that it extends
- MetaModel: A metamodel that extends other metamodel with new modeling elements



Phases for Modeling Extensions Construction

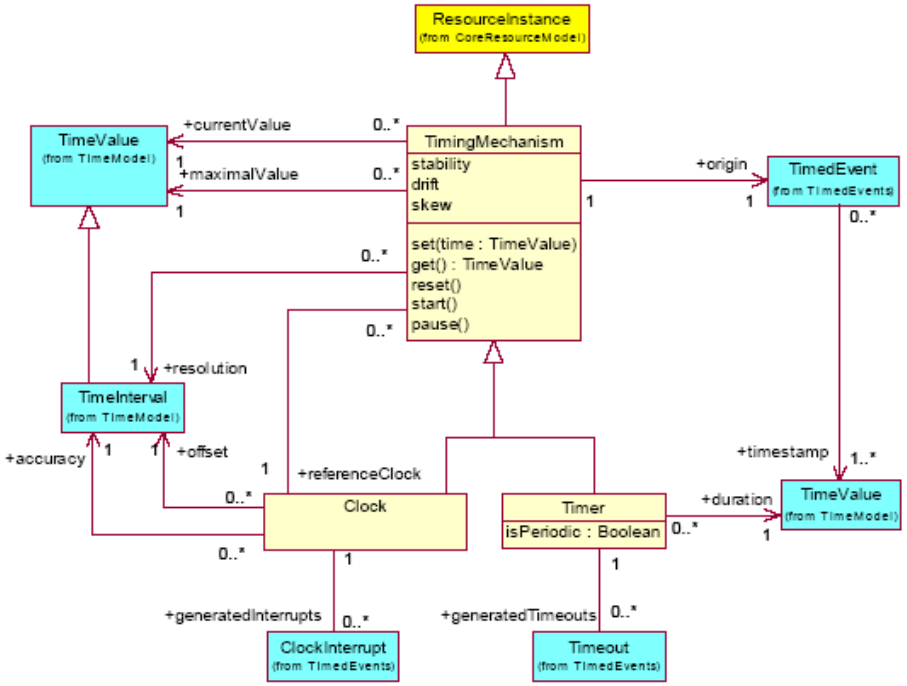
- **Conceptual Models**: A conceptual model is a simplified representation of new concepts that the models can include
- **Metamodels**: Abstract Syntax and Semantic. The metamodel defines a new modeling language that, often, is integrated with other metamodels
- **Profiles**: Concrete Syntax. Approach for the description of modeling notations based on UML model elements
- **Mapping From UML+Profile to Metamodel**: Integration of semantics, combination of two extensions methods
- **Transformations**: the automatic transformation reuses the software development knowledge.

Conceptual Models

- A **conceptual model** is an **analysis model** to identify the concepts to be included in the new language
- It includes concepts extracted on the basis of an **analysis of the domain, technique or technology** to be supported
- Often, technological extensions (e.g. middleware interface description languages) are well specified and do not requires this model
- We need a conceptual model, when the concepts of the extension are imprecise

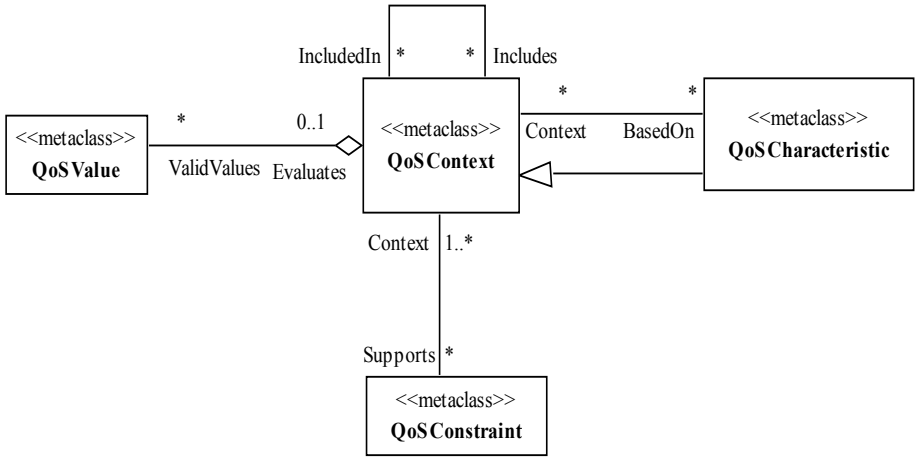
Conceptual Models Examples

- UML Profile for Scheduling Performance and Time is an example of standard that includes conceptual models for the description of extensions
- There are many approaches to consider timing concepts. This profile represent aspects of time based on:
  - TimeInterval
  - Clock and ClockInterrupt
  - Timer and Timeout
  - TimeValue and TimeEvent



MetaModels

- The Metamodel defines the **modeling elements** that represent the concepts that include the conceptual model
- It represents the **abstract syntax** of a modeling language



- The metamodel includes the **OCL well-formed rules**
- The new modeling elements can make reference to other metamodels
- MOF models specify the metamodels

MetaModels

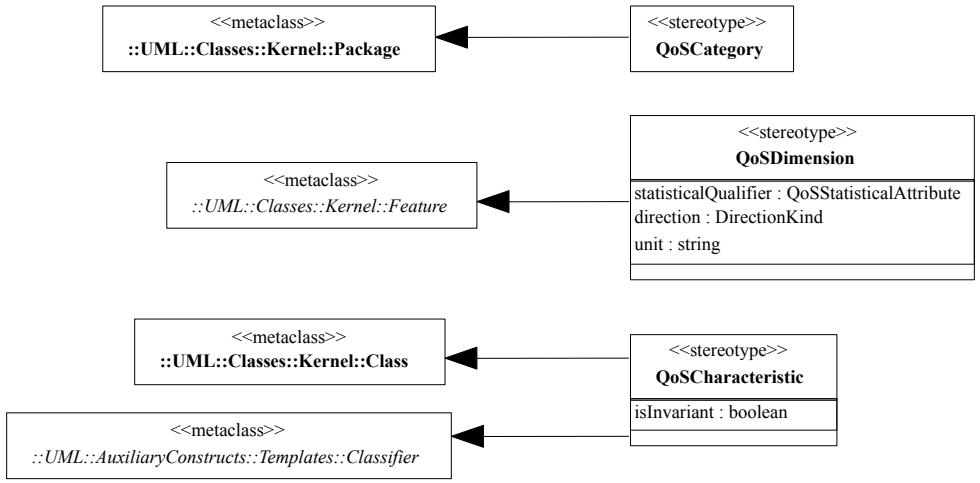
- Metamodels are the inputs for the construction of repositories in MetaModeling Frameworks (e.g. EMF)
  - They provides the automatic generation of repositories of new abstract syntax
- We have a repository, but we haven't yet support for the concrete syntax of our new modeling language (e.g. graphical notations)
- We can follow two approaches in the construction of concrete syntax:
  - Profiles: we reuse UML modeling elements for the concrete representation
  - Graphic Editor Framework for the construction and representation of concrete syntax, editors, and representations

MetaModels

- Metamodels are the inputs for the construction of repositories in MetaModeling Frameworks (e.g. EMF)
  - They provides the automatic generation of repositories of new abstract syntax
- We have a repository, but we haven't yet support for the concrete syntax of our new modeling language (e.g. graphical notations)
- We can follow two approaches in the construction of concrete syntax:
  - Profiles: we reuse UML modeling elements for the concrete representation
  - Graphic Editor Framework for the construction and representation of concrete syntax, editors, and representations

UML Profile

- UML profiles represent the modeling elements included in metamodels based on **extended UML modeling elements**
- It represents a **concrete syntax** of modeling language based on UML



- The profile includes the **OCL well-formed rules**. This depends on metamodel well-formed rules and UML modeling extension
  - We reuse well-formed rules defined in metamodel, but the profile must take into account semantic of UML elements

Stereotypes

- A stereotype is a model element that defines additional values, additional constraints, and optionally a new graphical representation.
- Stereotypes augment the classification mechanism based on the built in UML metamodel class hierarchy
- Stereotype attributes specify new kinds of properties that may be attached to model elements.
- Slots represent the actual stereotypes properties of individual model elements
  - simple datatype values or
  - references to other model elements

Constraints

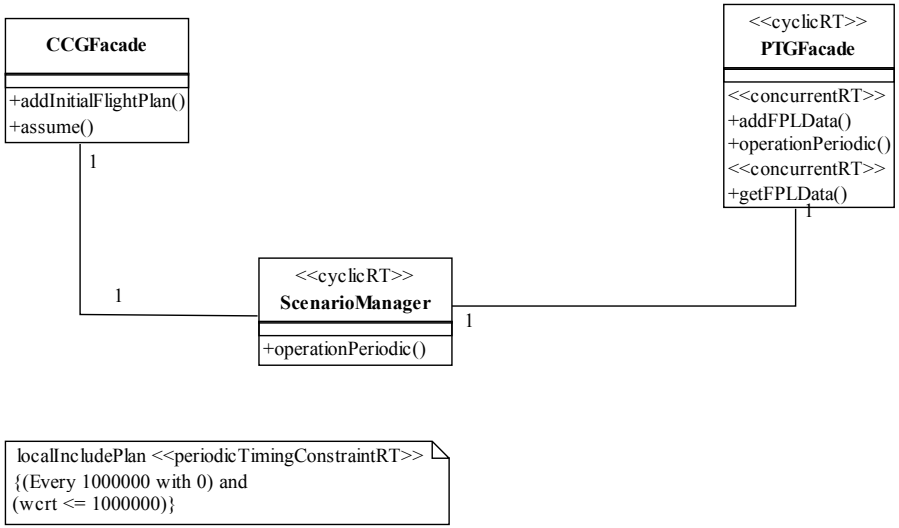
- Application constraints provide additional information of modeling concepts
  - They improve the semantic in structural models
    - They restrict allowed objects at MOF level M0
  - They improve semantic of some modeling elements such as provided/required interfaces
  - They define initial values of attributes and returned values in operations

Model Libraries

- Model library is a package that contains model elements that are intended to be reused by other packages
- A Model library does not extend the metamodel using stereotypes and attributes definitions
- A Model library is analogous to a class library in some programming languages
- Example:
  - Java, C++, ... Profiles (or model configurations) of some UML tools include in the model standard interfaces and classes of these languages to be used in the model
  - Libraries of reusable patterns
- Some UML extensions combine stereotypes extension and model libraries

Combination of Extension Approaches

- Stereotyped Constraints: Specific types of constraints

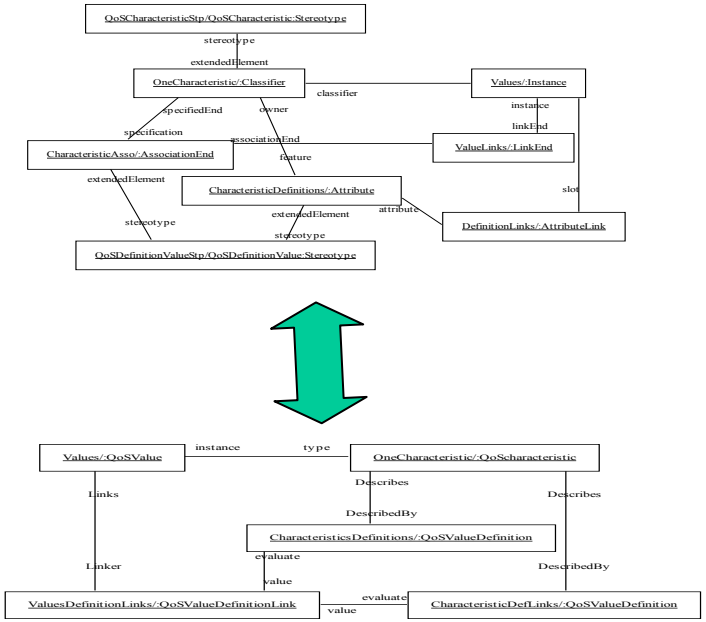


- Stereotyped Model libraries: Semantic of model libraries are improved with the stereotypes

Mapping From UML+Profile to Metamodel

- UML+Profile and Metamodel are two modeling languages that represent the same concept
- This mapping specifies how to express modeling elements of metamodel based on UML+profile
  - This is mapping between abstract syntax and concrete syntax based on profiles

UML Profile:
Modeling Language
UML + Extensions

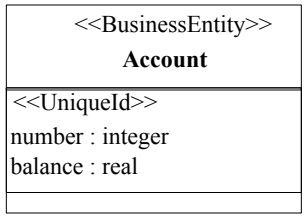


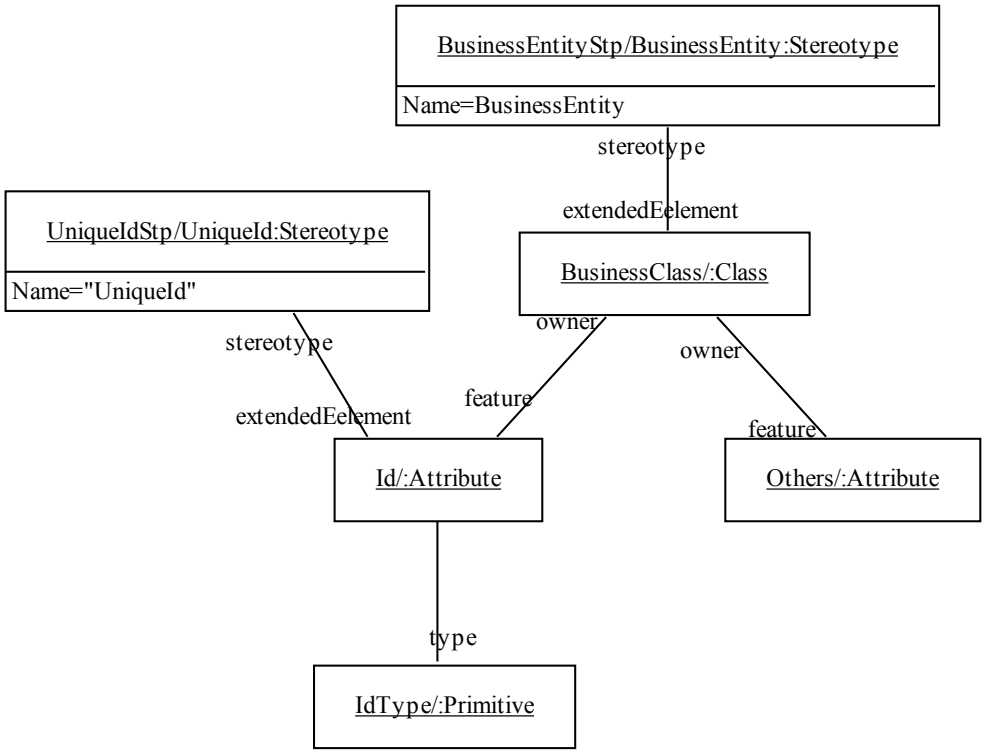
Meta-Model:
Significant Concepts
of Domain Problem

Specification of Mapping Profile to Metamodel (1)

- QVT is the approach for the precise description of this mapping
- When there is not a direct relation between metamodel and profile high level specific can provide a high level description of mapping
- A high level specification defines the equivalence between modeling concepts represented in UML+Profile and the metamodel language
  - Set of rules define the equivalence of same concept in two modeling languages
- UML+Profile models include instances of UML metamodel and profile slots

Specification of Mapping Profile to Metamodel (2)

- What is the Expression to represent  ?



- A collaboration of UML metamodel roles and OCL constraints describe the construction of a concept in UML
- A collaboration of metamodel roles and OCL constraints describe the same concept in the new language
- Both collaborations are the description of same concept in two languages