Towards Multi-modeling for Domain Description

Andrea Leitner

Christian Kreiner

Institute for Technical Informatics
Graz University of Technology, Austria
andrea.leitner@tugraz.at
Introduction

• Domain model representation
  – long lifecycle
  – collection of domain knowledge (make knowledge explicit)
  – core part of the product line

• Two main domain modeling paradigm
  – Domain specific language
  – Feature-oriented domain analysis
Outline

• Motivation – 2 examples
• Basic concepts
• Prototypical implementation
Example 1, Hybrid vehicle control unit

• **Different variants of hybrid electrical vehicles:**
  – Different vehicle types
  – Topology
    • Mild hybrid
    • Full hybrid
  – Transmission
    • Manual
    • Automatic
    • CVT
Representation with FODA

Major problems:

- Building „feature groups“
- not intuitive
- redundant information
- hidden information
Hybrid vehicle drivetrain topology

**Characteristics:**

- Few specialized elements
- Focus on architecture instead of functionality
- Several instances of components.
Example 2 – Fish farm
Problems with single model approach

• Choosing one approach
  – DSL vs FODA (different characteristics)
  – FODA:
    • basically everything can be described in terms of features
    BUT...
    for some characteristics a DSL fits better.
Multimodeling

- Combine feature-oriented and graphical representation.

- Graphical representation consists of
  - elements
  - connections between elements
Multimodeling - Concept

- Define constraints/relations between different models.
- Intra-model constraints are defined with built-in means.
- Application models are defined as usual.
Technical realization

- **Common Domain model**
  - realized as Feature Model

- **Configuration model**
  - mapping from DE models to corresponding AE models

- **Configuration space**
  - in pure::variants used to configure a concrete variant
  - use a pure::variants configuration space to automatically check constraints.

- **Definition of constraints:**
  - hasElement
  - hasAttributeValue
  - hasConnection
Constraint definition

- "hasElement" – constraint
  - checks the existence of an element
Constraint definition

• „hasAttributeValue“ – constraint
  – checks the value of a specific attribute
Constraint definition

- "hasConnection" – constraint
  - checks if a connection between two elements exists.
Check constraint

• Create a new `configurationSpace` containing the `CommonDomainModel`.

• Start constraint checking mechanism on multimodel variant.
  – *go through all elements in application model*
  – *check if element exists in external model and select/unselect element.*
  – *if element has an attribute – get current attribute value from external model.*
  – *if element == connection – check if connection between the two elements exists.*

• Validity of the application model is checked automatically by `pure::variants`
Future work

• Implement solution space integration
  – code generators as pure::variants Family Model.
• Parsers for other tools (e.g. MetaEdit+)
Summary

• Combine feature-oriented and graphical representation
• Use of existing technologies
• Enhanced representation of knowledge to
  – support understanding
  – improve readability
Thanks for your attention!

Questions?