Design Patterns for Metamodel Design

*Domain-Specific Modeling Workshop*
Portland, Oregon
October 23, 2011

Hyun Cho and Jeff Gray

University of Alabama
Department of Computer Science

This work supported in part by NSF CAREER #1052616.
Domain-Specific Modeling Languages

- Customized to a specific domain
- Designed and implemented by domain-driven needs and abstractions
- Components of a DSML

Diagram:
- DSML
- Abstract Syntax (Metamodel)
- Conform
- Symbolized
- Constrained
- Semantics
- Concrete Syntax
- Models
Domain-Specific Modeling Languages (cont.)

- Benefits of DSMLs
  - Easier to learn and provide rich expressiveness
  - Evidence that DSMLs increase productivity and improve quality
- Several success stories, but not widespread adoption
- What is the reason? Perceived reasons by some:
  - High development cost and complex development process
  - Lack of vendor support
  - Lack of robustness
  - Lack of standardization
  - Requires both domain knowledge and language development expertise

The quality of DSMLs largely depends on a designer’s domain experiences and language development expertise
Metamodels can be designed (or inferred) by reusing existing metamodel concepts that represent commonly recurring metamodel design issues across multiple domains.

Such reuse of metamodeling experience may improve the quality of metamodel design as well as achieve a significant increase in productivity in the development of DSMLs.
Approach for Identifying Metamodel Design Patterns

- Collect various types of DSMLs
- Identify characteristics of DSML and its modeling elements
- Analyze commonality of DSMLs
- Identify candidate metamodel design problems
- Collect and review metamodel samples
- Propose metamodel design patterns
## Collection of DSMLs Examined

<table>
<thead>
<tr>
<th>Domain</th>
<th>Diagrams</th>
<th>Brief Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent Discrete Event System Modeling</td>
<td>Petri Net</td>
<td>Modeling systems with concurrency and resource sharing</td>
</tr>
<tr>
<td>Data Modeling</td>
<td>ERD</td>
<td>Model the logical structure of database</td>
</tr>
<tr>
<td>Project Management</td>
<td>Gantt Chart</td>
<td>Model project activities with relevant information (i.e., duration, cost, ...)</td>
</tr>
<tr>
<td>Project Management</td>
<td>PERT Chart</td>
<td>Identify the critical path of the project by modeling the sequence of tasks</td>
</tr>
<tr>
<td>Electronic Circuit Design</td>
<td>Schematic Diagram</td>
<td>Represent how electronic components are connected with others</td>
</tr>
<tr>
<td>Electronic Circuit Design</td>
<td>PCB Layout</td>
<td>Show the placement of electronic components on printed circuit board</td>
</tr>
<tr>
<td>Molecular Modeling</td>
<td>-</td>
<td>Model the structures and reactions of molecules</td>
</tr>
<tr>
<td>SW Design</td>
<td>Flowchart</td>
<td>Model process or algorithm</td>
</tr>
<tr>
<td>SW Design</td>
<td>Component Diagram</td>
<td>Represent static structure of components and their relations</td>
</tr>
<tr>
<td>SW Design</td>
<td>UseCase Diagram</td>
<td>Describe system functionalities or behaviors with UseCase and Actor</td>
</tr>
<tr>
<td>SW Design</td>
<td>Class Diagram</td>
<td>Describe the static structure of the system in terms of classes</td>
</tr>
</tbody>
</table>
Identify Characteristics of DSMLs

- Context Diagram
Identify characteristics of DSMLs

- Electronic Circuit Design: Palm III Charger

Schematic Diagram

PCB Layout Diagram

Images are copied from http://www.harbaum.org/till/palm/cradle/index.html
Identify characteristics of DSMLs (cont.)

- Component Diagram

Images are copied from http://www.agilemodeling.com/artifacts/componentDiagram.htm
Identify characteristics of DSMLs (cont.)

- Component Diagram
Identify characteristics of DSMLs (cont.)

- UseCase Diagram

Image is copied from http://www.agilemodeling.com/artifacts/useCaseDiagram.htm

Copyright 2004 Scott W. Ambler
Features of DSMLs

Composition Rule
- Containment and/or Nesting may be required when a classifier contains another classifier and/or is organized hierarchically
- Style and Boundness are required to describe the semantics of Relationship
Questions for Identifying Candidate Patterns

- What could be a primitive or base metamodel pattern, which could be common ground for metamodel design?

- How to extend the base metamodel if a DSML has complicated language constructs?
  - For example, a DSML can have typed relationships such as include and extend in UseCase diagram?

- How to represent boundedness in metamodel?

- How to design the metamodel to describe containment and nesting?
References for Metamodel Design

- Web Pages
Base Metamodel Pattern

- What could be a primitive or base metamodel pattern, which could be common ground for metamodel design?
- How to represent *boundedness* in metamodel?
- Applicable for simple Box-and-Line style DSMLs
  - Most common pattern for early stage of DSML development
  - Useful for Prototyping DSML

![Diagram](a) Modified BPMN p92

- Evaluation Points
  - none
Metamodel with (sub)types Pattern

- How to extend the base metamodel if a DSML has complicated language constructs?
- Extension of base metamodel design pattern
  - Add more expressiveness to DSMLs
  - Semantics of each relationship is required to enforce behaviors and properties

**Evaluation Point**
- Association point between Classifiers and Relationships
Containment/Nesting Pattern

- How to design the metamodel to describe *containment* and *nesting*?
- Some DSMLs may contain or nest modeling elements to control the abstraction level
  - Can focus on core thoughts by eliminating unnecessary details or give more descriptions by showing details

![Diagram of Containment/Nesting Pattern]

(a) Modified BPMNp119  
(b) Excerpted from UML Infrastructure V 2.3 p87  
(c) Modified from Q-ImPrESS Project p49 & BPMN p100

- Evaluation Point
  - Comprehensibility and Extendibility
Application of Metamodel Design Patterns

- Composition-based metamodel development

Typed Relationship  Classifier Containment
Metamodel Inference

- Metamodel can be inferred from a set of model examples through grammar inference.
- To infer accurate metamodel, a large set of domain examples, both positive and negative examples.
- But, preparing a large set of domain examples are practically difficult.
- Metamodel design patterns can be used as a common sample data.
Expected benefits of Metamodel Design Patterns

- Avoid duplication of metamodel design for recurring design problems
- Keep high quality metamodel fragments
- Guide and Recognize key patterns and best-practices of metamodel design
- Reduce time-to-market for developing new DSMLs
Thank you for your attention