THE UNIVERSITY OF ALABAMA
COMPUTER SCIENCE

Design Patterns for Metamodel Design

Domain-Specific Modeling Workshop

Portland, Oregon

October 23, 2011

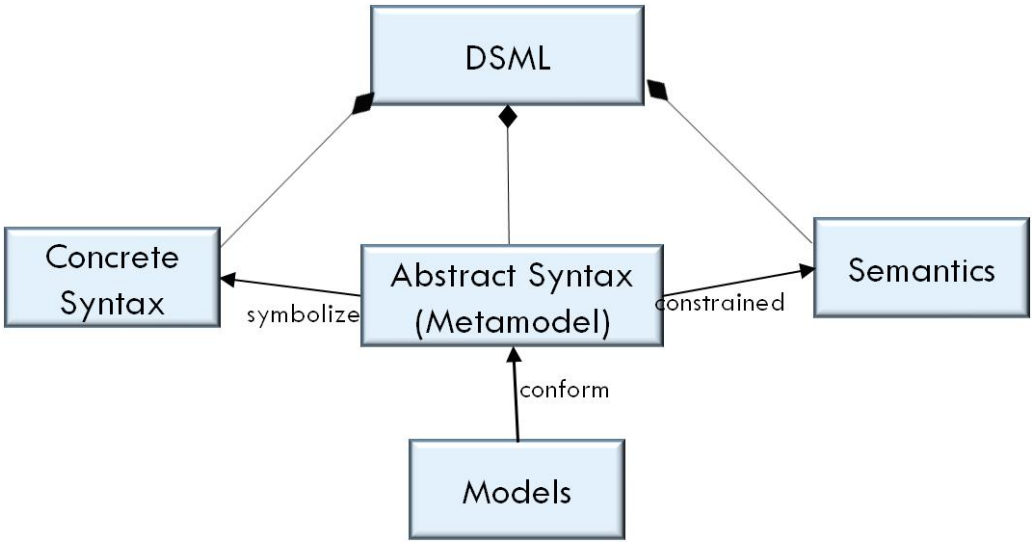
Hyun Cho and Jeff Gray

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



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

Research Hypothesis

- *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

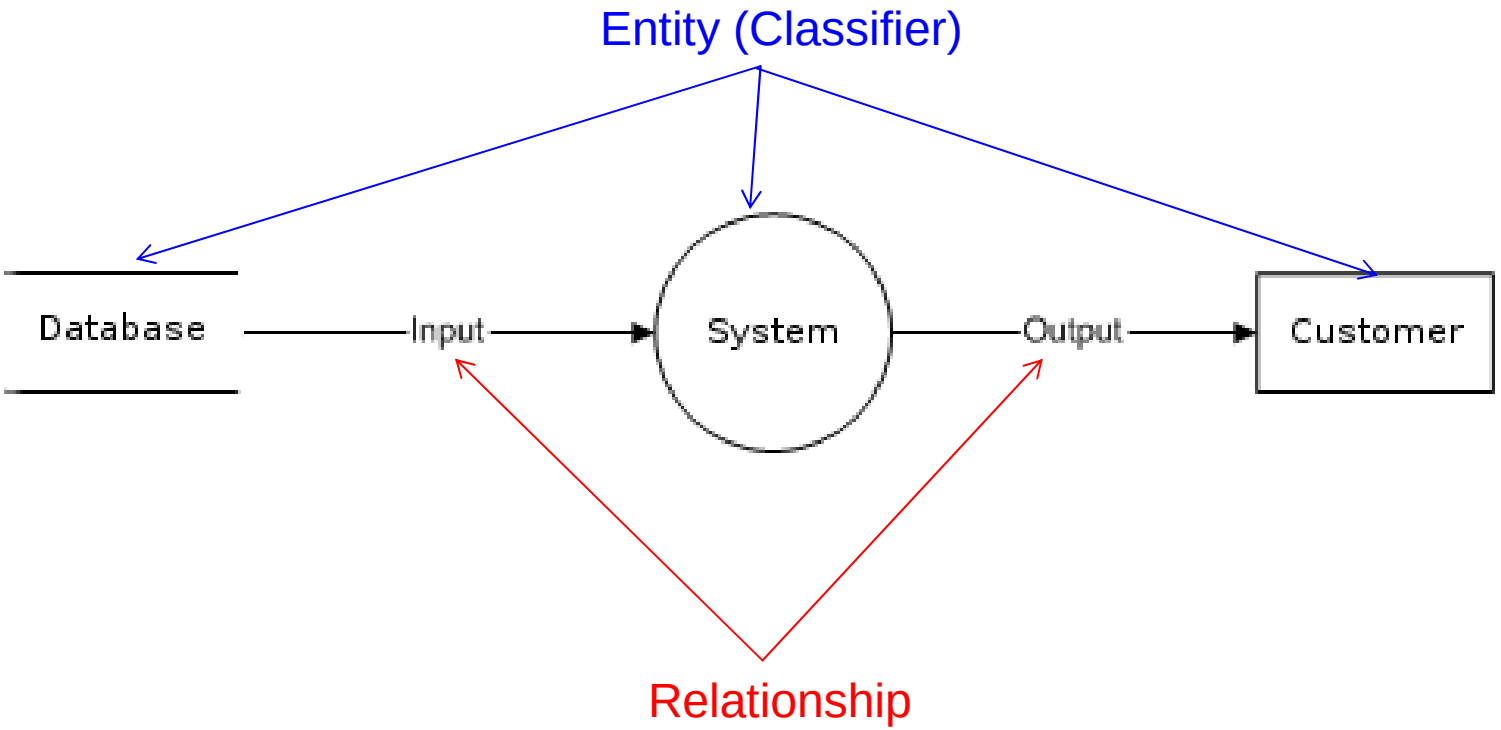
- Collect various types of DSMLs
- Identify characteristics of DSML and its modeling elements
- Analyze commonality of DSMLs
- Identify candidate metamodel design problems
- Collect and review metamodel samples
- Propose metamodel design patterns

Collection of DSMLs Examined

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

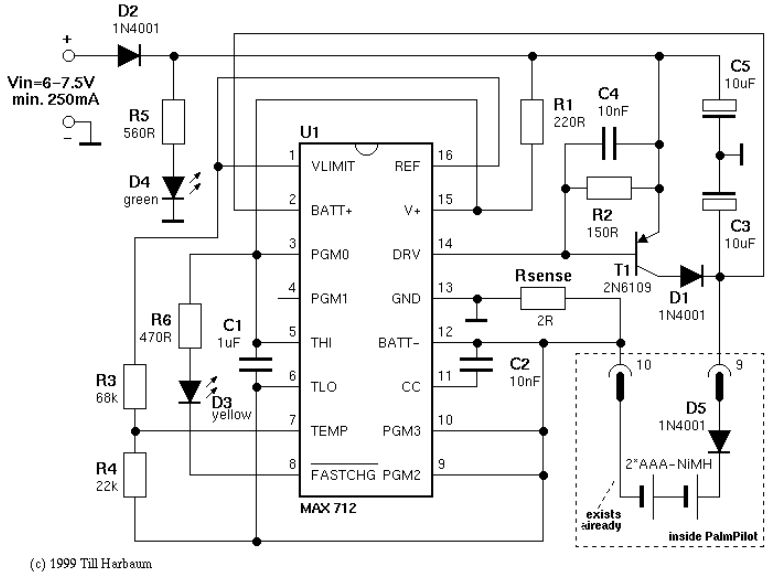
Identify Characteristics of DSMLs

- Context Diagram



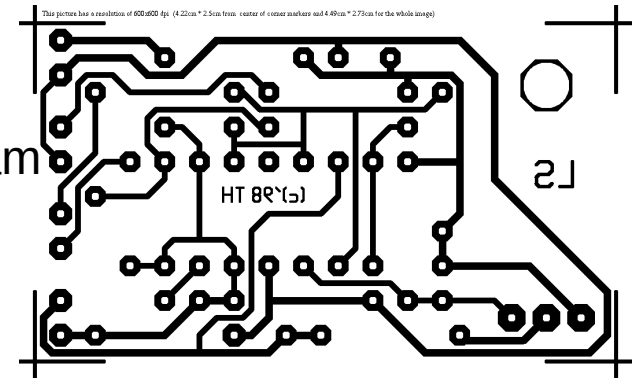
Identify characteristics of DSMLs

- Electronic Circuit Design: Palm III Charger



Schematic Diagram

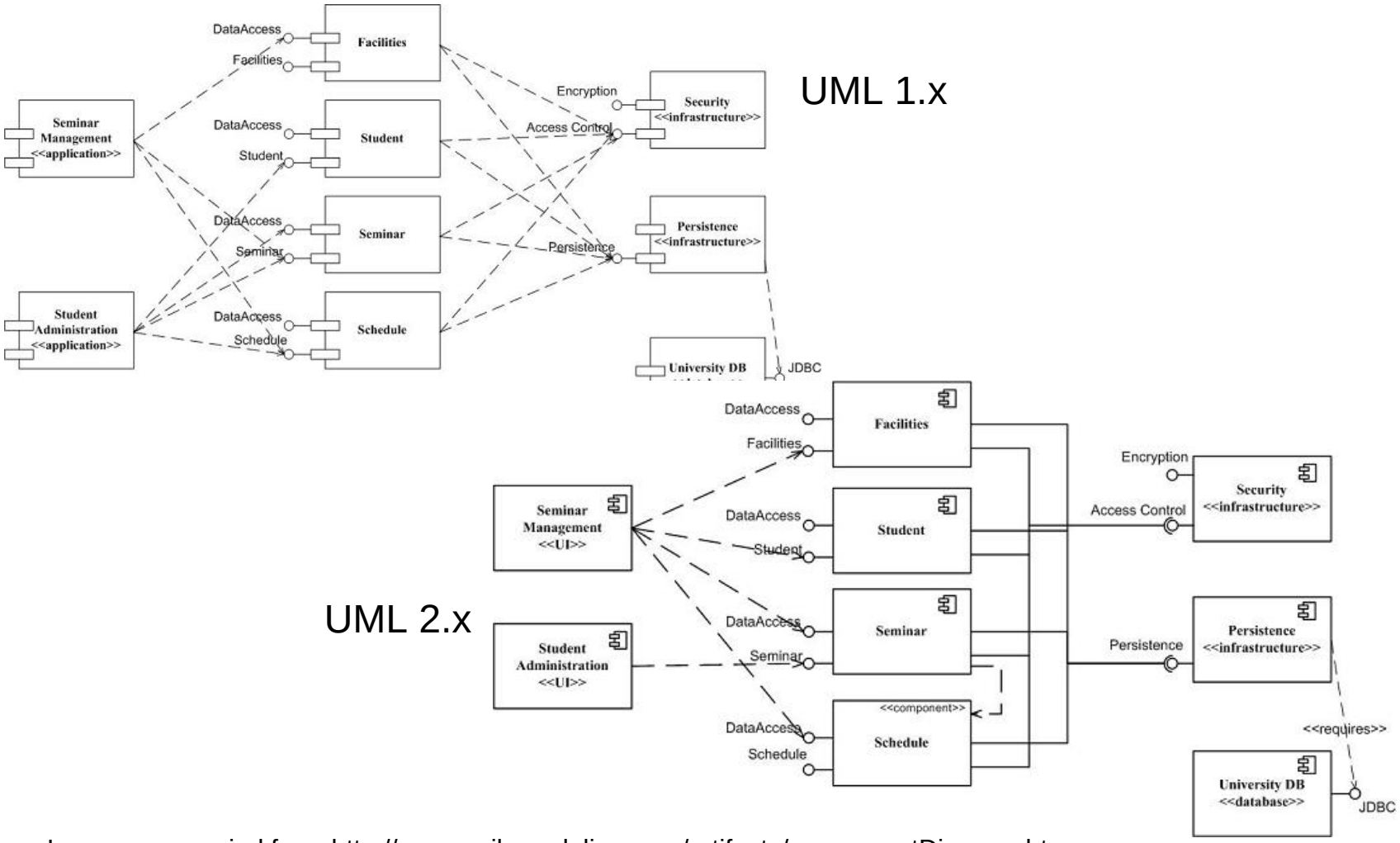
PCB Layout Diagram



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

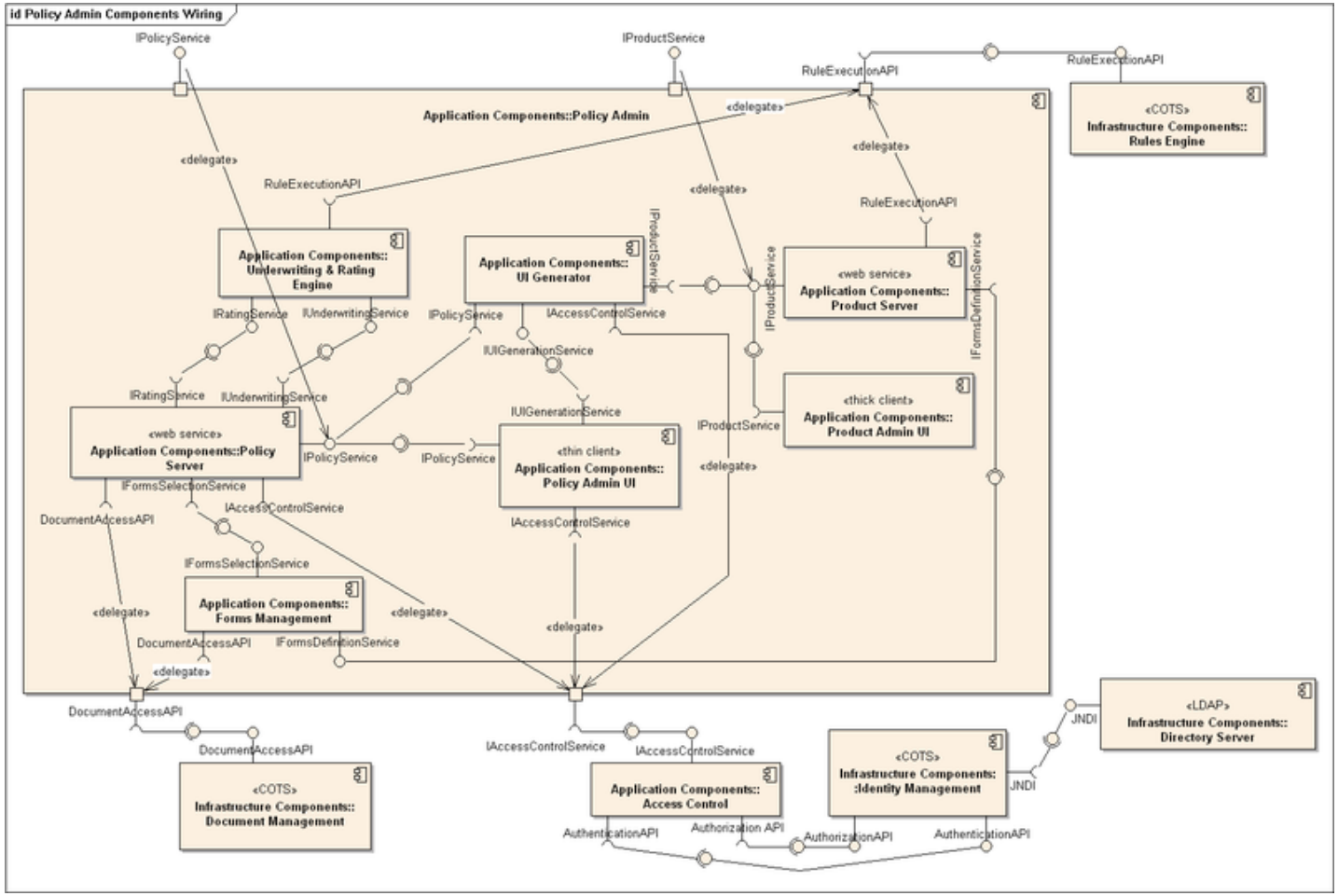


Image is copied from http://blogs.uis.edu/group8/files/2011/07/800px-Policy_Admin_Component_Diagram.png

Identify characteristics of DSMLs (cont.)

- UseCase Diagram

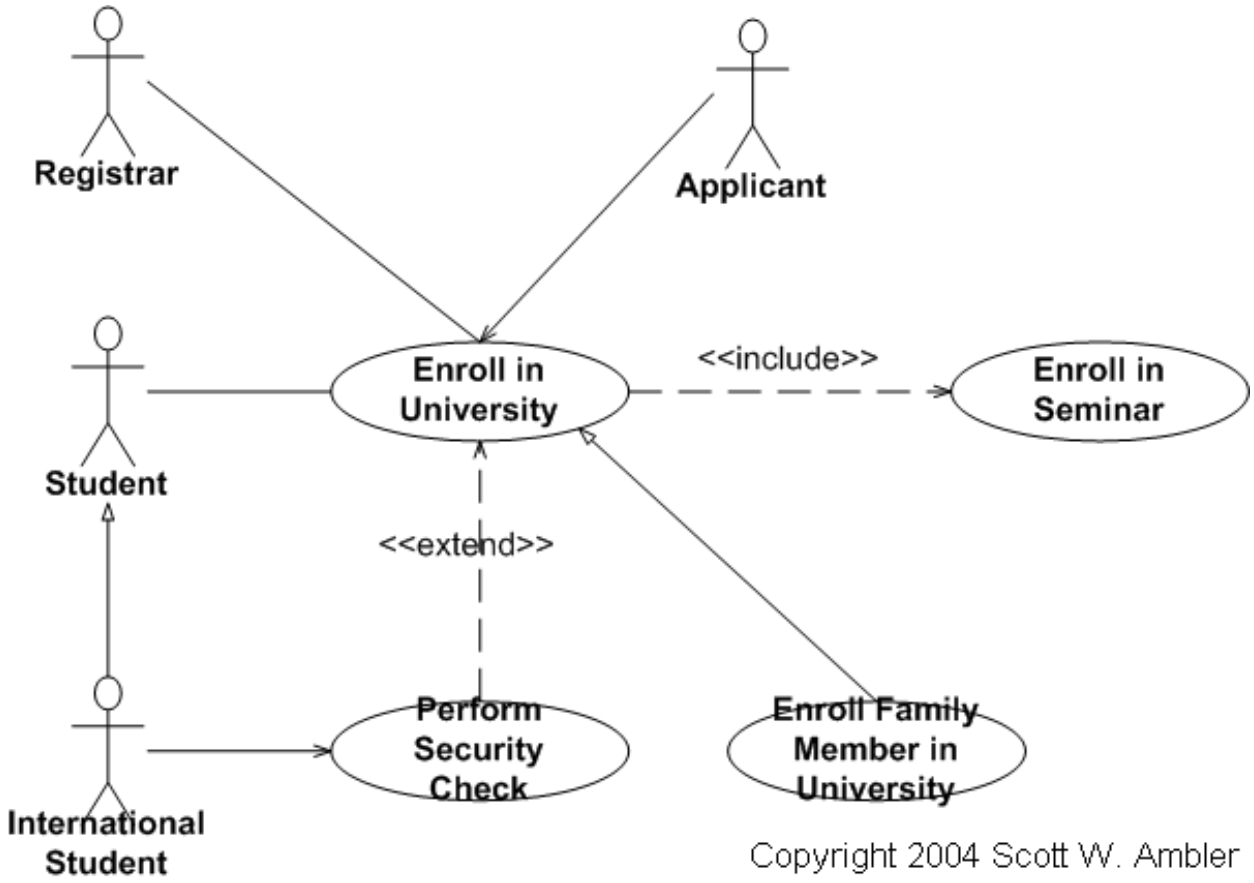
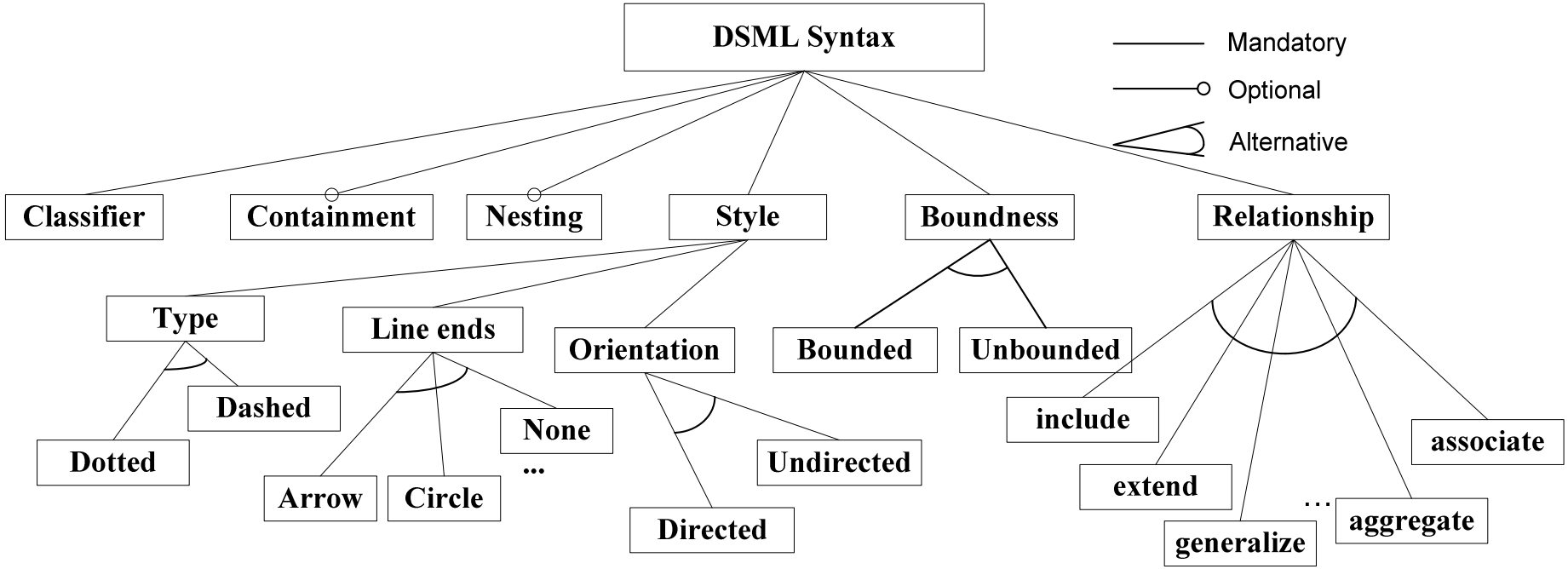


Image is copied from http://www.agilemodeling.com/artifacts/useCaseDiagram.htm

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 Boundedness are required to describe the semantics of Relationship

Questions for Identifying Candidate Patterns

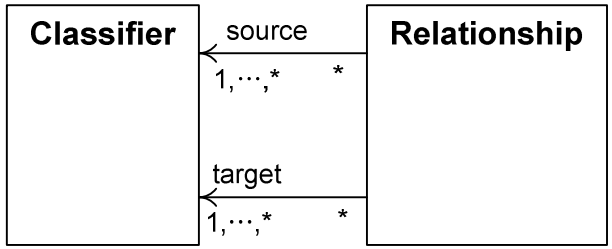
- What could be a primitive or base metamodel pattern, which could be common ground for metamodel design?
- How to extend the base metamodel if a DSML has complicated language constructs?
  - For example, a DSML can have typed relationships such as include and extend in UseCase diagram?
- How to represent *boundedness* in metamodel?
- How to design the metamodel to describe *containment* and *nesting*?

References for Metamodel Design

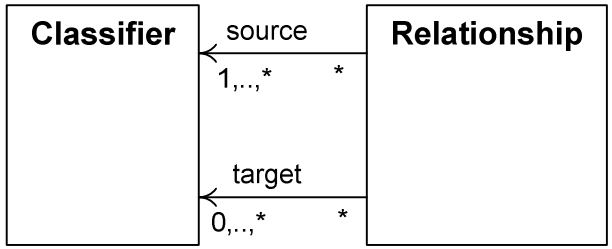
- OMG UML 2 Superstructure, http://www.omg.org/spec/UML/2.3/Superstructure/PDF
- OMG Business Process Model And Notation (BPMN) Ver. 2.0, http://www.omg.org/spec/BPMN/2.0
- QImPrESS Service Architecture Meta-Model, http://www.q-impress.eu/wordpress/wp-content/uploads/2009/05/d21-service_architecture_meta-model.pdf
- Ouardani, A., Esteban, P., Paludetto, M., & Pascal, J. C. 2006. A Meta-modeling Approach for Sequence Diagrams to Petri Nets Transformation within the requirements validation process. In Proceedings of the European Simulation and Modeling Conference, pp. 345-349, Toulouse, France
- 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



(a) Modified BPMN p92

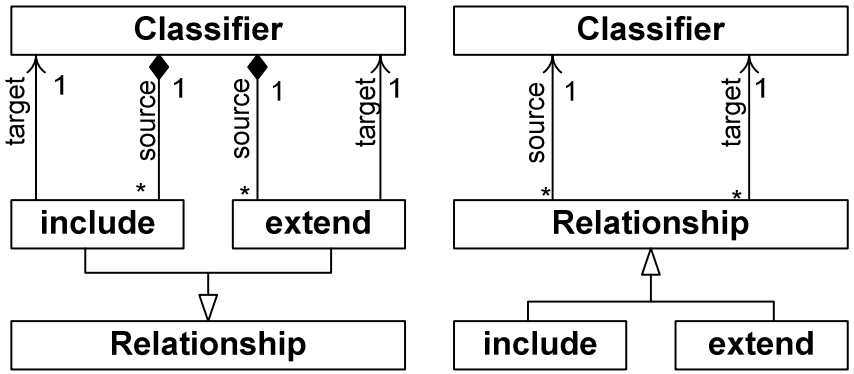


(b) Modified Bender et el. metamodel design

- Evaluation Points
  - none

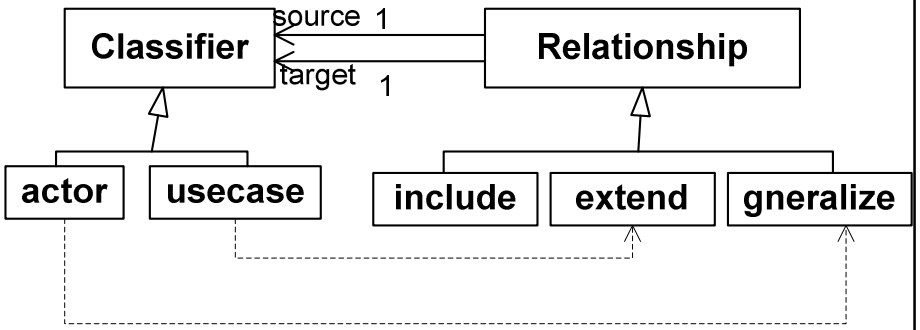
Metamodel with (sub)types Pattern

- 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



(a) Excerpted from UML Superstructure V 2.3 p620

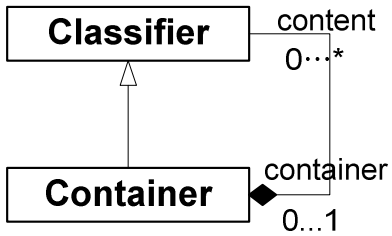
(b) Simplified Ouardani et al. metamodel design



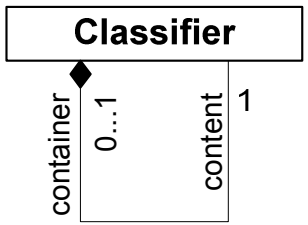
- Evaluation Point
  - Association point between Classifiers and Relationships

Containment/Nesting Pattern

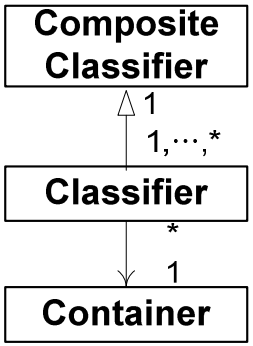
- How to design the metamodel to describe *containment* and *nesting*?
- Some DSMLs may contain or nest modeling elements to control the abstraction level
  - Can focus on core thoughts by eliminating unnecessary details or give more descriptions by showing details



(a) Modified BPMNp119



(b) Excerpted from UML Infrastructure V 2.3 p87

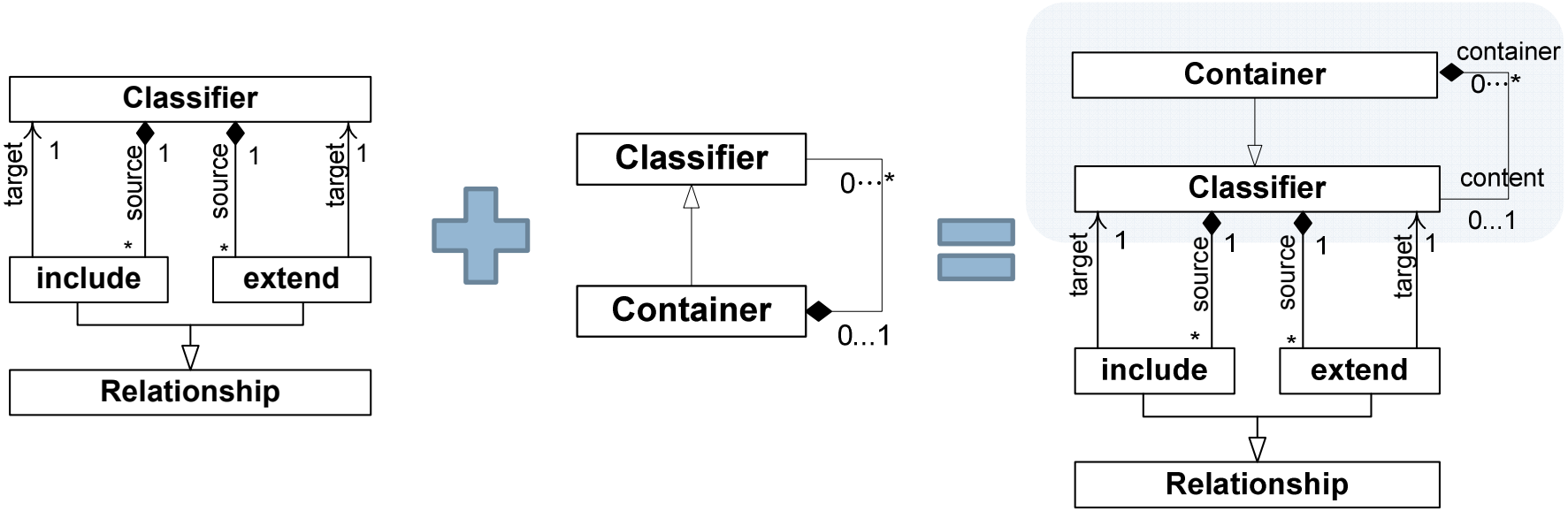


(c) Modified from Q-ImPrESS Project p49 & BPMN p100

- Evaluation Point
  - Comprehensibility and Extendibility

Application of Metamodel Design Patterns

- Composition-based metamodel development



Typed Relationship

Classifier Containment

Application of Metamodel Design Patterns

- 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

This work supported in part by NSF CAREER #1052616.