SWE 721 / IT 821 Reusable Software Architectures

Static Modeling for Software Product Lines

Hassan Gomaa
Department of Information and Software Engineering
George Mason University

Reference: Hassan Gomaa, Chapter 6 in "Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures", Addison-Wesley Object Technology Series, 2005

Static Modeling for Single Systems

- Static Model
  - Define structural relationships between classes
  - Depict classes and their relationships on class diagrams
- Relationships between classes
  - Associations
  - Composition / Aggregation
  - Generalization / Specialization
- Static Modeling during Analysis
  - System Context Class Diagram
    - Depict external classes and system boundary
  - Static Modeling of Entity classes
    - Persistent classes that store data

Static Modeling for Software Product Lines

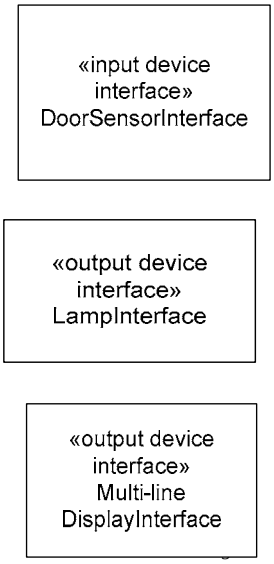
- Static Modeling of SPL problem domain
  - Real-world classes
    - Physical classes
    - Entity classes
  - SPL context class diagram
    - Define boundary of SPL
  - SPL entity class diagrams
    - Model information intensive SPL classes

UML Modeling for Software Product Lines

- Depict class and object categorization using UML stereotypes
- **Stereotype** defines
  - New modeling element derived from existing UML modeling element
  - Tailored to modeler's problem
  - Depicted using guillemets
    - «entity», «interface», «control»
- UML 1.4 upwards supports multiple stereotypes for class
  - Use UML stereotypes to depict reuse category
  - Use UML stereotypes to depict application role category

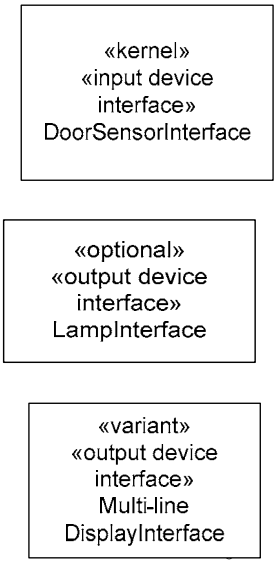
Static Modeling for Single Systems

- UML 1.4 upwards supports multiple stereotypes for a modeling element
- Single systems (COMET)
  - Categorize each class by application role using stereotype
    - «control», «entity», «interface»
  - Object has same application role stereotype as class it is instantiated from



Static Modeling for Software Product Lines

- UML 1.4 upwards supports multiple stereotypes for a modeling element
- Single systems
  - Categorize each class by application role using stereotype
  - «control», «entity», «interface»
- **Software Product Lines (PLUS)**
  - **Second UML stereotype depicts reuse category**
    - **«kernel», «optional», «variant»**



Static Modeling of SPL Problem Domain

- During Analysis Modeling
  - Conceptual static model
  - Emphasizes real-world classes in the problem domain
  - Does not initially address software classes
  - Emphasis on
    - Physical classes
      - Have physical characteristics (can see, touch)
    - Entity classes
      - Data intensive classes
- **Figure 6.2 Conceptual static model for Microwave Oven System**
- **Figure 6.3 Conceptual static model for Microwave Oven Product Line**

Software Product Line Context Model

- Defines boundary between software product line and external environment
  - Depicted on UML product line context class diagram
  - System Context Class Diagram for a family of systems
- Software Product Line
  - Consider as one aggregate class
  - <<product line system>>
  - Represents any member of SPL
- Model external entities to SPL as external classes
  - SPL boundary is variable

Software Product Line Context Model

- Model each external class using two stereotypes
  - E.g., Fig. 6.1
- Role categories of external classes (Fig. 6.4)
  - <<external I/O device>>
  - <<external user>>
  - <<external system>>
  - <<external timer>>
- Reuse categories of external classes
  - <<kernel >>
  - <<optional>>
  - <<variant>>

Associations on SPL Context Class Diagram

- SPL Context Class Diagram shows
  - Association between product line system and external class
  - Multiplicity of association (1 to 1, 1..* to 1, etc.)
  - Kernel external classes
    - 1 to 1, 1..* associations with product line system
  - Optional, variant external classes
    - 0..1, 0..* associations with product line system
- Associations have standard names
  - «external input device» Inputs to «product line system»
  - «product line system» Outputs to «external output device»
  - «external user» Interacts with «product line system»
  - «external system» Interfaces to «product line system»
  - «external timer» Awakens «product line system»

Software Product Line Context Model

- Development strategies
  - Forward evolutionary engineering
- Kernel first approach
  - Context model for kernel developed first
    - Depict kernel external classes
      - Figure 6.5 Microwave Oven product line Kernel System context class diagram
- Software Product Line Evolution
    - Consider optional and variant external classes
      - Figure 6.6 Microwave Oven product line context class diagram

Software Product Line Context Model

- Development strategies
  - Reverse evolutionary engineering
- View Integration Approach
  - Develop context model for each member of SPL
  - Integrate context models -> Product Line context model
    - External classes common to all members -> kernel external classes
    - External classes only in some views -> optional external classes
    - Determine alternative (variant) external classes
- Software Product Line Evolution
  - Consider further evolution of context model
- Example: Figs 15.17 – 15.20 SPL Context Model for Factory Automation SPL

Static Modeling of Entity Classes

- Entity classes
  - Data intensive classes
  - Store long-living (persistent) data
  - Especially important for Information System SPLs
    - Many are database intensive
  - Also important for many real-time and distributed SPLs
- During analysis modeling
  - Model entity classes in the problem domain
  - Attributes
  - Relationships

Entity Class Models for Software Product Lines

- Kernel first approach
  - Entity class model for kernel developed first
- Software Product Line Evolution
  - Consider optional and variant classes
- View Integration Approach
  - Develop entity class model for each member of SPL
  - Integrate entity class models -> SPL entity class model
  - Classes common to all members -> kernel classes
  - Classes only in some members -> optional classes
  - Classes with differences

    -> Specialized variant subclasses of generalized superclass

Figs 6.8-6.12 Entity class model for E-commerce SPL

Object & Class Structuring Criteria

- Determine all software objects and classes in SPL
  - Use Object Structuring Criteria
  - Guidelines for identifying objects
- Structuring criteria depicted using stereotypes
  - «entity», «interface», «control»
- Objects are categorized (Figure 6.1)
  - Interface object
    - Interfaces to and communicates with external environment
  - Entity object
    - Long living object that stores information
  - Control object
    - Provide overall coordination for collection of objects
  - Application Logic Object
    - Responsible for executing application specific rules or algorithms

Object Structuring Criteria

- Interface object
  - User interface object
    - Interfaces to a human user
  - Device interface object
    - Interfaces to an external device
  - System interface object
    - Interfaces to external system or subsystem
- Entity object
  - Data abstraction object
    - Encapsulates data structure
  - Database wrapper object
    - Hides details of access to DBMS

Object Structuring Criteria

- Control object
  - Coordinator object
    - Decision making object, not state dependent
    - Decides when, and in what order, other objects execute
  - State dependent control object
    - Defined by finite state machine
      - Statechart or state transition table
  - Timer object
    - Activated periodically

Object Structuring Criteria

- Application Logic Object
  - Business Logic Object
    - Encapsulates business rules
  - Algorithm Object
    - Encapsulates problem domain algorithm
  - Agent object
    - Encapsulates knowledge of application domain