Evolutionary Process Model for Software Product Lines
Software Application Engineering

- Software Application
  - Member of software product line
- Software Application Engineering
  - Derive application architecture from SPL architecture
- Select application features subject to
  - Feature dependencies and relationships
- Derive software application architecture
  - Kernel components always selected
  - Optional and variant components correspond to features selected

Software Application Engineering Phases

- Application Requirements Modeling
- Application Analysis Modeling
- Application Design Modeling
- Incremental Application Implementation
- Application Testing
- Figures 12.1, 12.2
Software Application Engineering
Application Requirements Modeling

• Define functional requirements of application
  – Application Feature and Use Case Models
• Select application features subject to
  – Feature dependencies
  – Feature relationships
    • Mutually exclusive (0 or 1) feature group
    • One-and-only-one feature group
    • One-or-more-of from feature group
  – Decide on values of parameterized features
• Given Feature / Use Case Table
  – Determine application use case model
    • All product line kernel use cases
    • Some optional use cases
    • Some alternative use cases subject to feature constraints

Software Application Engineering
Application Analysis Modeling

• Develop application static and dynamic models
• Application Static model
  – Given feature/class dependencies, determine
    • Application context model
    • Application entity class model
• Application Dynamic Model
  – Based on selected features and use cases
    • Select communication diagrams
      – Based on selected use cases
      – Selected variation points
        » Use default or variant objects
• Application Statechart model
  – Selected statecharts for selected state dependent objects
  – Feature conditions determine
    • Feature dependent states, transitions, actions
Software Application Engineering
Application Design Modeling

• Given
  – Software product line architecture
  – Selected features
  – Feature/Class model
• Adapt product line architecture
• To derive application architecture
  – Determine application components

Software Application Engineering
Incremental Application Implementation

• Select subset of application based on use cases
  – Need components that realize use cases
  – Some previously implemented components will exist in SPL Repository
    • E.g., kernel components
    • Optional or variant components developed for previous application
  – Some new components will need to be implemented
    • E.g., optional, variant components not previously developed
    • Detailed design, code, unit test of new components
Software Application Engineering

Application Testing

- Incremental Application Integration
  - Integration testing of each application increment
    - Integration test based on use cases
  - Develop integration test cases for each use case
  - White box testing
    - Test interfaces between objects in use case
- Application System Testing
  - Includes functional testing of system
    - Testing of functional requirements
  - Black box testing
    - Based on use cases
- Need system test for each increment released to user
- Independent test team
  - Goal is to break system
  - Thorough systematic test of system before release to users

Application Deployment

- Define component instances
  - For components that can have multiple instances
  - Define component parameters
- Interconnect component instances
  - As given by application architecture
    - E.g., Fig. 12.4
- Map component instances to physical nodes
  - Depict physical configuration of target application on
    - UML Deployment Diagram
    - E.g., Fig. 12.5
Tradeoffs in Software Application Engineering

- Problems in Application Engineering
  - Application requirement may not be supported by
    - SPL feature model or use case model
  - Needed application component is not compatible with SPL architecture
- Systematic approach
  - Iterate through Evolutionary SPL Process
  - Adapt SPL feature and use case models
  - Evolve analysis and design models
  - Derive new application
- Pragmatic approach
  - Adapt application models without changing SPL models
  - Application architecture no longer compatible with SPL architecture

Example of Software Application Engineering

- Microwave Oven Application
- Feature model (Fig. 12.6)
- Use case model (Fig. 12.7)
  - Determine from feature / use case model (Table 12.1)
- Static Application Context model (Fig. 12.8)
- Dynamic Communication model (Figs. 12.9, 12.10)
  - Based on selected features
- Feature/Class Dependency (Table 12.2)
- Microwave Oven Application Classes (Figs. 12.11, 12.12)
- Application Software Architecture (Fig. 12.13, 12.14, 12.15)
- Application Deployment Diagram (Fig. 12.16)