FORM : A feature-oriented reuse method with domain-specific reference architectures

Kyo C. Kang, Sajoong Kim, Jaejoon Lee, Kijoo Kim, Euiseob Shin and Moonhang Huh


2010-08-18
Park Bum Seok
Contents

- Introduction
- Background
- Related work
- Feature oriented reuse method
  - Domain engineering
  - Application engineering
- Conclusion
- Discussion
Introduction (1/2)

❖ **Reuse** of software artifact
  - One of the most promising solutions to the “software crisis”

❖ General software reuse environment

**Diagram:**
- Pool
  - Library A
  - Component A
  - Component B

**Process:**
1. Create
2. Store
3. Search
4. Use

**Questions:**
- How to make reuse systematically and effectively?

**Observations:**
- No specification to create reuse modules
- Expect to be reused sometime by someone
- No specification for using reuse module

© KAIST SE LAB 2010
Introduction (2/2)

- Previous research for systematic reuse
  - Feature oriented domain analysis (FODA)
    - Focus on creating reusable assets
    - No systematic method how to use reusable assets

- Research goal
  - Introduction of effective and explicit software reuse method and process extends FODA
    - Method to create reusable software artifact effectively
    - Method to develop new application from reusable artifacts

  Feature Oriented Reuse Method (FORM)
Feature

Definition

- Informally essential characteristics of applications in a domain
  - Services or functions provided by the system
  - Domain technologies
  - Implementation techniques
- From other paper
  - An end user visible characteristic of a system
  - Distinguishable characteristic of a concept that is relevant to some stakeholder of the concept
- Genus of feature is ‘characteristic’

Abstraction that both customers and developers understand can be used for requirement engineering

Two groups: Functional and non-functional features
Domain

- Definition
  - A family of related systems

Domain analysis

- Definition
  - Examining a family of related systems and extracting the ‘commonalities and differences’ of these systems
  - Feature oriented domain analysis
    - Doing domain analysis based on features

Domain engineering

- Definition
  - Using analysis results to create a set of reference models
  » Creating reusable software architectures and components
Feature model

Definition
- The model constructed during the analysis which represents relationships between features as an AND/OR graph

Introduced from FODA

Characteristics
- Communication language among stakeholders
- Easy to capture commonality & difference of applications on feature oriented models
- Representative of application

< Feature model >
Related work

- **FODA**
  - **Purpose**
    - Introduce development method that creates reusable assets
  - **Feature model is introduced & take a key role to identify commonalities and to generate architecture model**
  - **Limitation**
    - No systematic method about how to use reusable assets

- **FORM**
  - **Purpose**
    - Introduce method of creating a set of reusable assets and also developing new application from the assets
  - **Feature model is revised and upgraded to cover non-functional features** and to support detailed architecture
How to create artifacts effectively for reuse?

- Issues for high reusability
  - The more commonality
  - The more accurate specifications (e.g. interface, scope)
  - The more architectural information (e.g. where to use)

- Proposed approaches for this issues
  - Applications in a same domain can have more commonalities
    - Domain-specific
  - To extract commonalities easily can make artifacts more reusable
    - Feature-oriented
  - Matching information between architecture and component and
    Standard architectures are helpful for reuse
    - Reference architecture and component
Feature Oriented Reuse Method (FORM)

- Systematic method that focuses on capturing commonalities and difference of applications in a domain in terms of “features” and using the analysis result to develop domain architectures and components.
FORM (3/11)

Engineering process

FORM Domain Engineering

- Domain Analysis
- Feature Model
- Reference Architecture Development
- Reference Architecture
- Reusable Component Development
- Reusable Component

FORM Application Engineering

- User Requirement Analysis
- Feature Selection Spec
- Application Architecture selection
- System Model
- Process Model
- Module Model
- Application Software Development
- Application Software

Process Product Model reuse Feed back

< FORM engineering process >
Domain engineering

**Purpose**
- Analyzing systems for commonalities and differences
- Creating feature model
- Creating reference architectures and reusable components

**Three phases**
- Context analysis
  - Identification of scope of domain and intended use of the domain
- Domain (or feature) modeling
  - Identifying user understandable features on target domain
  - Constructing ‘feature model’ that represents features’ interrelationships during analysis
- Architecture (or component) modeling
  - Construction of reference architectures and components
- Revised feature model
  - For non functional feature
- Four layers
  - Capability or Service
  - Operating environment
  - Domain technology
  - Implementation technique
FORM feature model (Cont’d)

- Roles of FORM feature model
  - Standardizing both meaning of features and communication
  - Evaluating commonalities and differences between domains, and applications in a domain
  - Standardizing to create reusable domain artifacts
  - Standardizing development of application using domain artifact

<Feature model comparison>

Application A

Feature model A

Easily compare

Feature model B

Application B

Feature model C

Application C
Domain analysis

- **Purpose**
  - Identifying commonalities and differences of systems in a domain
  - Constructing ‘feature model’ according to the analysis

- **Process**
  - Planning
    - First identification of family of products in a domain
  - Feature analysis
    - Identifying features, classifying them, organizing them
    - Constructing ‘feature model’
  - Validation activities
    - Validating feature model before using
Architecture modeling

**Purpose**

- Construction of reference architecture and reusable component

**3 levels of abstraction of reference architecture**

- **Subsystem model**
  - Defining overall system structure by packaging service features
  - Applying localization of function blocks

- **Process model**
  - Decomposition of subsystem into set of processes
  - Applying software principles of loosely coupled, highly cohesive
  - Considering environment features

- **Module model**
  - Defining module model based on the features on domain technology and implementation technique
  - Containing specification how to integrated into application
Feature model and reference architecture

Non functional features

Capabilities
- services, operations, presentation, use, attributes, etc

Operating Environment
- H/W platform, O/S, DBMS, network, etc

Domain Technologies
- Method, theories, etc

Implementation Technologies
- Algorithms, Abstract data types, etc

Subsystem model
- Packaging of services & operations

Process model
- Refinement considering performance characteristics

Module model
- Abstraction and Specification of Module component

Feature Model © KAIST SE LAB 2010 < Mapping features to architecture models >
Application engineering

- Definition
  - Finding a correct reference architecture through feature selection and then putting in reusable software components
Feature selection

Definition
- Finding a matching set of features from the feature model

Effective feature selection by following the four-level feature hierarchy
- Capability, operating environment, and finally domain technologies and implementation techniques

Order of feature selection
1. Analyzing user requirements for the target application
2. Constructing feature model for the target application using stepwise refinement comparing with reference feature models
3. Selecting closest feature mode and use the corresponding reference architectures
Conclusion

❖ Contribution
- FODA is extended in FORM to cover the entire development process including reuse process
- Feature model is effectively upgraded for designing, architecting and reusing

❖ Future work
- Formalize language that formal analysis is possible
- Feature model will be extended to support development of application optimized for specific problem context
Discussion

❖ Pros

- Focus on reuse concept and introduce practical method

❖ Cons

- Meaning of feature is not clearly fixed among researches
- FORM is semi formal
  - Provide construction language but not rigorous analysis of models
Thank You.