INF329

Presentation of

Alaña, Rodríguez (2007) Domain Engineering Methodologies Survey (GMVSA 20580/07, GMV, July 2007, 38 pp.)

by Anne Elise Weiss

2012-02-06

Last time

- Domain Engineering in general
- Definition of a domain
- Some DE methods
- Problems wrt DE + OOA/D
- Some OOA/D methods supporting DE

Purpose and scope of the report

- Identify methodologies
- Identify and select language for generic models
- Summarize different models, notations and tools
- Analyse compliance of GMV-proposed approach with ISO 12207

Methodologies

Divides DE into three phases:

- Domain analysis
- Domain design
- Domain implementation

Work to be carried out according to the SOW:

- Perform a DE analysis through a variation analysis of space systems
- Perform a DE design to be reused for design and development of future space crafts

Methodologies

The phases:

1. Domain analysis
Discovers and formally describes the commonalities and variabilities. Output: domain model; explicit representation

Domain model consists of:

- Domain dictionary – defines terms
- Context models – specify boundaries
- Feature models – hierarchical decomposition of features

Methodologies

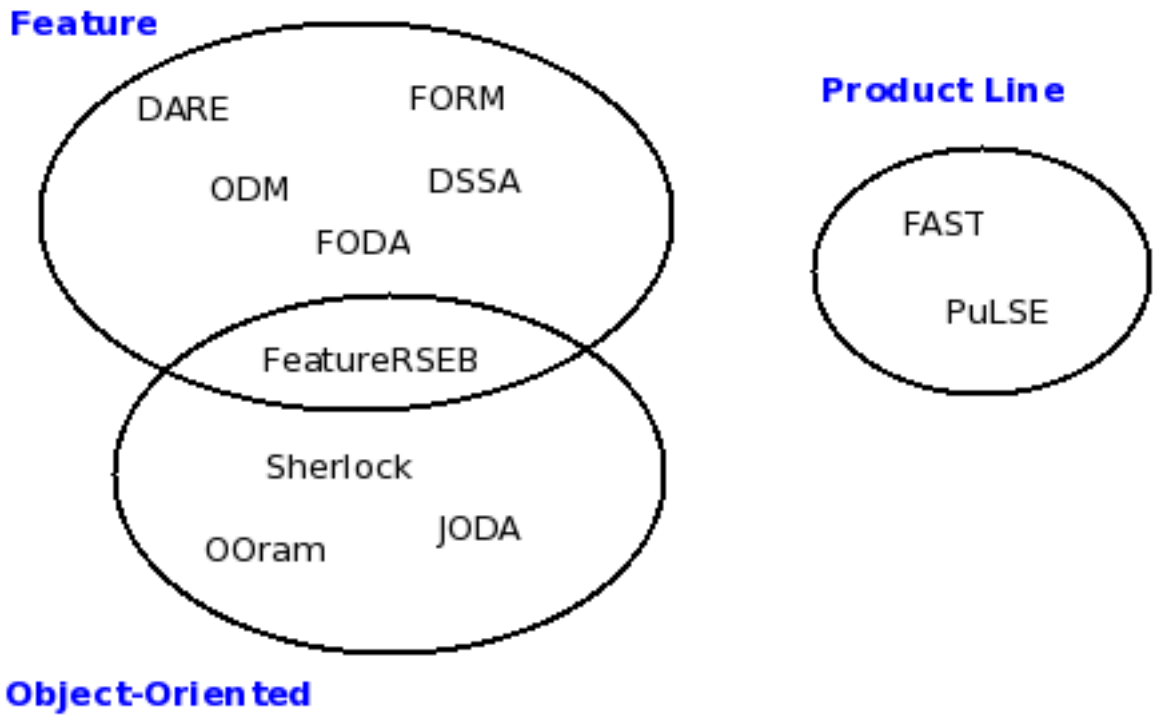
2. Domain design

- Input: Domain model
- Applies a partitioning strategy to produce a generic design
- Defines different elements and how the domain features are allocated to them

3. Domain implementation

- Input: Design models and generic architectures
- Main output: reusable assets, application generations, domain languages

Main groups of DE methods



Methods based on the analysis of the domain

ODM (Organization Domain Modeling)
Developed to provide an overall framework for a DE life cycle.

Divides DE into three phases:

- Plan the domain – "Domain of focus" based on stakeholders' interests
- Model the domain – document domain information, produce domain model
- Engineer an asset base – sub-phases to scope, architect and implement

Methods based on analysis of the domain

ODM (Organization Domain Modeling)

Advantages:

- Useful for a wide range of organisations and domains
- May be integrated with a variety of SE processes, methods and implementation technologies

Disadvantage:

- No support for creating DSL and application generators

Most successful: mature, stable, economically viable domains.

Methods based on analysis of the domain

FODA (Feature-Oriented Domain Analysis)

- Based on identification, analysis and documentation of the main features
- Result: generic domain products based on abstraction

Three phases:

1. Context analysis – Establish bounds of domain, relation with other domains
    - Result: structure and data-flow diagrams
2. Domain modelling – Analyse context model to generate domain models

Methods based on analysis of the domain

FODA (Feature-Oriented Domain Analysis)

Three phases:

3. Architecture modelling – create high-level architecture model from the domain model

- FODA's features represented hierarchically --> simple to identify and understand feature model
  - Component-based development not encouraged, but can be achieved using Object Connection Architecture
- FODA has no specific process for req. spec, verification and management

Methods based on analysis of the domain

FORM (Feature-Oriented Reuse Method)

- Extends FODA to the design phase
- Uses feature model to develop domain architectures and components
- Reason: Features + code should be packed, managed and reused as software modules

Three phases:

- Context analysis: identify scope and interaction
- Feature modelling: commonalities and variabilities. Hierarchical feature diagram
- Architecture/component modelling: define a set of reference architectures

Methods based on analysis of the domain

FORM (Feature-Oriented Reuse Method)

- Reference architectures are defined using the feature model
- Organized in three hierarchical levels
  - Subsystem model (system structure)
  - Process model (dynamic behaviour)
  - Module model (set of features)

- The modules are basis for generation of reusable components.
- Mapping between feature and architectural model is needed.

Methods based on analysis of the domain

FeatureRSEB (Feature Reuse-Driven Software Engineering Business)

- Process that has integrated the feature modelling of FODA into processes and work products of Reuse-Driven Software Engineering Business (RSEB).
- RSEB: Use-case driven systematic reuse process based on UML.

RSEB DE activities:

- Application Family Engineering (higher level)
- Component System Engineering (lower level)

Methods based on analysis of the domain

FeatureRSEB

- FeatureRSEB developed because RSEB is based on modelling variability, but doesn't include DA techniques or description of a systematic way to perform the asset development
- Combines FODA and ODM concepts
- Feature models are simpler than FODA's
- Architecture + reusable subsystems: use-case diagrams, transformed into object models

Methods based on analysis of the domain

FeatureRSEB

- Includes DA, solving the limitations of RSEB
  - DA starts with domain scoping and feature modelling
    - Components
    - High level use case model
  - Next: identification of commonality and variability of the elements.
    - Use case and object model: domain entities and the interaction between them
    - Sequence and interaction diagrams: dynamic relations among the domain entities

Methods based on analysis of the domain

DSSA (Domain-Specific Software Architectures)

- Architecture for a specific domain based on commonalities and differences
- Focus on the process: how to define these features and derive the final architecture

- Domain analysis:
  - capture components and operations in a class of similar systems in a particular domain
  - define relationships + data and control flow
    - result: requirements document
  - Identify constraints and requirements

Methods based on analysis of the domain

DSSA (Domain-Specific Software Architectures)

- Develop architecture
- Last step: Develop reusable components based on the architecture and information

Methods based on analysis of the domain

Sandwich method

- Specifies components that can be implemented and put into a library
- Domain models: generic architecture or standard designs
- Low-level components act as building blocks --> reuse guaranteed

Domain analysis:

- Domain information, entities, models, expand and verify models and classification
- Result: Domain model

Methods based on analysis of the domain

Sandwich method

Domain model includes:

- Concepts to enable specification of systems
- Plans describing how to map specifications into code
- Rationales for the specification concepts

Two procedures:

- Bottom-up activities, low-level common functions. Products are associated with the structures derived by:
- Top-down activities, for system analysis. Result: generic architectures

Drawbacks:

- Little information related to the whole process
- No support for development of languages

Methods based on analysis of the domain

DARE (Domain Analysis and Reuse Environment)

- Support environment for partially automating the activities of domain analysis
- Focus: activities to acquire and structure knowledge
- Domain Analysis Book
- Domain must already be defined

Four activities (iterative):

1. Acquire domain knowledge
2. Structure domain knowledge automatically
3. Identify commonalities
4. Generate domain models

DE methods based on the product line

- Methods linked to product lines and software families
  - groups of products that share common features *and* meet the needs of a particular market area
- Few available methodologies

FAST (Family-Oriented Abstraction, Specification and Translation)

- Defines DE + Application Engineering process
  --> covers the whole product-line engineering process

Domain has to satisfy these requirements:

- mature
- stable
- experts must exist and be available

DE methods based on the product line

FAST

- Software family is defined
- Environment for producing family members is developed
- DE process is based on sharing common features

Two phases:

- Domain/commonality analysis
  - Collect/document knowledge
  - Define decision model
  - Design Application Modelling Language (AML)
- Domain implementation
  - Development and refinement of the specified environment

DE methods based on the product line

FAST

- No specific technique recommended
- When domain specified: translation into products is carried out using a DSL, so translation can be done automatically

DE methods based on the product line

PuLSE (Product Line Software Engineering)

Elements:

- Four deployment phases
  - Initialization
  - Infrastructure construction
  - Infrastructure usage
  - Management/evolution
- Technical components
- Support components

No recommended tool or technique for any of the activities.

DE and OOA/D methods

OOram (Object Oriented Role Analysis and Modelling)

- Provides a framework for creating a variety of methodologies
- Dev. cycle focused on interactions: improves reuse, traceability, complexity
- Idea: different methodologies needed for different purposes
- Defines a "role model", collecting objects together according to common goal

Three processes

- Model creation process
- System development process
- Reusable asset building process

DE and OOA/D methods

JODA (Joint Integrated Avionics Working Group Object-Oriented Domain Analysis)

- Uses OOA/D instead of functional methods for domain analysis
- DA: what is reusable, how can it be structured and reused

DA consists of three phases:

- Domain preparation
- Domain definition
- Domain modelling -- which extends from OOA/D
  1. Def. attributes and services, objects, relationships
  2. Domain scenarios
  3. Abstraction and grouping of objects

DE and OOA/D methods

Sherlock

- Product line practice
- Uses OOA for analysis
- Uses different diagrams for modelling
- Input: informal description of the domain
- Output: set of architectural models
- Tool support for managing each activity
- No specific technique for req.spec., verification, traceability

SODA (Strategic Options design and Assessment)

- Approach to design long-lived system architectures

Activities:

1. Develop strategic scenarios
2. Propose business strategies
3. Design architectural scenarios (result: proposed architectures)
4. Assess scenario feasibility

- Aim for the final result: flexible architecture that is adaptable to change over time

Architectural analysis methods

- Methods concerning the transition from the domain modelling to the domain architecture definition
- During the architectural analysis, the domain engineer selects an appropriate design approach for building a generic design
- Many methods only provide a high-level feature model --> a transition from the domain model to the final architectural design is needed

Architectural analysis methods

OCA (Object Connection Architecture)

- Architectural model used to structure a generic design
- Typically used with FODA
- Input: domain model, architecture information
- Result: generic design, used in application development

Mapping process:

1. Analysis of the domain model
2. Process of mapping objects and subsystems onto code templates

DE notations and tools

- During DA: the bounds of a domain are identified
- These bounds have to be represented somehow

Some notations linked to features:

- SADT (Structured Analysis and Design Technique): describing systems as a hierarchy of functions
- UML (Unified Modelling Language): semi-formal, object-oriented
- SysML (System Modelling Language): based on UML 2.0, DSML for system engineering applications.

The generated reusable assets: output to XML files

DE notations and tools

Many domain analysis methodologies are based on features, so tools that can model features are also needed. E.g.:

- Xfeature – supports feature modelling, uses standard technology
- RequiLine – requirements engineering tools for management of software product lines
- pure::variants – commercial tool, supports feature modelling and configuration

DE notations and tools

Notations to represent the domain design:

- SysML
- UML2 – supports MDA and MDD
- AADL (Architecture Analysis and Design Language) – provides features for modelling a software system's conceptual architecture

Tools for developing domain design:

CORBA, Eclipse framework, MS Visio, TOPCASED

DE notations and tools

Some notations supporting domain implementation

- MDA – provides a framework to MDD.
  - Basic function: generation of applications from a set of procedures (UML model) --> mechanism to transform the feature model instances into an executable application automatically or semi-automatically.

- HRT-UML (Hard-Real Time UML Models) – defines an extension profile of UML.
  - Used to model generic architectures, especially useful for modelling hard real-time systems.

Conclusion

Conclusion regarding the most suitable DE approach:
Feature-Oriented Domain Analysis (FODA).

Reasons:

- FODA represents the domain knowledge using several complementary models
- Oriented towards commonalities and variabilities
- Easily understandable feature models (end user + designer)
- The method is generic
- Tight relationship between FODA-generated models and those found in the majority of OOA/D

Conclusion

Reasons:

- FODA specifies the whole DA process until the architecture design
- Has been applied to several industry application domains with good results