Object-Oriented Domain Engineering: Introduction and Overview

by
Charles McKay,
Software Engineering
University of Houston-Clear Lake
Recommended Prerequisites

To gain maximum benefit from this tutorial, participants should:

• have an interest in the concepts, issues, and potential of domain and application engineering

• be familiar with concepts and techniques of object-oriented modeling of systems
Recommended Prerequisites Continued

- The tutorial will introduce domain engineering

- The tutorial will briefly review and then apply object-oriented modeling to the analysis stages of domain engineering
Outline

• Overview of Domain Engineering Concepts
  - Contrast with Point Solution Approach
  - Engineering Perspectives
  - Some Anticipated and Reported Benefits
  - Some Relevant Definitions and Types of Models
  - Contrasting Scenarios
  - Introduction to Example Problem 1
Outline Continued

- Key Features of the Object Paradigm Used in This Approach
  - Object Classes and Relationship Sets
  - Object Models, Views & Product Sets
  - Outline of Transformation Team’s Analysis
  - Example Problem 1
  - Introduction to Example Problem 2
- Summary
- References
- Acknowledgements
Overview of Domain Engineering Concepts
Contrast with Point Solution Approach

Point Solution Approach
• Business-as-usual
  - customer has a problem that requires a computing system as part of the solution

  - customer wants the solution ASAP and ‘chooses’ to sacrifice long term maintenance and operation costs in favor of an earlier, ‘less expensive’ development that meets immediate needs and requirements
Contrast with Point Solution Approach Continued

- This commitment to a quick, low cost, development ‘fix’ has often resulted in sacrificing opportunities to:

  - ‘design for reuse’ and to

  - ‘design to facilitate safe, affordable and timely adaptation to meet changing requirements in the future’
Contrast with Point Solution Approach Continued

- i.e., long-term, life cycle costs and benefits are not the major project drivers--(e.g., often 90% of life cycle costs are in post-deployment)
Contrast with Point Solution Approach Continued

Domain and Application Engineering Approach

• The customer and/or developer recognize an application domain (i.e., a particular type of problem space such as ‘embedded control systems for motor vehicles’) that will require several instances of an application solution (i.e., there will be many automobiles that will need embedded control systems)
Contrast with Point Solution Approach Continued

• Such application domains often contain distinct subdomains that can benefit from a ‘product line’ which provides a solution to this distinct subset of the problem space

  - e.g., an anti-lock braking system, a cruise control system, a fuel-injection control system
Contrast with Point Solution Approach Continued

- Note that such product lines often have interdependencies (e.g., stepping on the brake pedal will disengage the cruise control system while engaging the anti-lock braking system)

- Note the opportunities to maximize reuse!
Contrast with Point Solution Approach Continued

Domain engineering studies and models the

• ‘common’ requirements and constraints of the problem space (i.e., the ‘Domain Model’)

• ‘generic’ solution architectures for the product lines of this ‘Domain Model’ (i.e., a ‘Domain Specific Systems and Software Architecture’)
Contrast with Point Solution Approach Continued

- implementation(s) of the components of the generic architectures
- maintenance of the domain model, the generic architectures and the component implementations
Contrast with Point Solution Approach Continued

Application engineering studies and models:

• the commonalities and differences of a particular application in the domain from the models created by domain engineering (e.g., the anti-lock braking system for an 18 wheel tractor-trailer will have some different requirements and constraints from an anti-lock braking system for the more numerous instances of passenger automobiles)
Contrast with Point Solution Approach Continued

• the adoption and possible tailoring of reusable components from the models of domain engineering plus the development of new components specific to the application (80/20 %?)

• the maintenance of the application-specific components
Engineering Perspectives
Domains, Product Lines and Engineering Perspectives

Domain with Multiple Product Lines

- HW Eng
- SW Eng
- Sys Eng
- Life Cycle Support Envir.
- Human Factors Eng
- HW Eng
- *
‘Domain’, ‘As Is’, & ‘To Be’ Models in Domain Analysis & Domain/Application Design

Applications
‘As Is’ Systems
Domain/Applications
Research
Related Technology
Research
‘To Be’ Systems
Req’s & Constraints

‘As Is’ Req’s & Cnstrt’s Outside Domain Model
→
In: Ap
Des

‘Domain’ Model: Common Req’s & Cnstrt’s for Applications in the Domain
→
Dom
Des

‘To Be’ Req’s & Cnstrt’s Outside Domain Model
→
In: Ap
Des
Some Anticipated and Reported Benefits
Anticipated Benefits of DA&DP

-- DA&DP: Domain Analysis and Design Process

• Minimize redundant software development efforts,

• Improve process productivity,

• Improve product quality in system/software development and maintenance life-cycles,
Anticipated Benefits of DA&DP

• Reduce costs and schedules,

• Increase user suitability,

• Reduce risks,

• Increase opportunities for competition by lowering domain knowledgebase barriers, and

• Facilitate technology insertion.
Reported Benefits of Domain Specific Reuse

• Common Ada Missile Packages (CAMP)
  - 1984 STARS (pioneering) program (Prieto 1992)

• Restructured Naval Tactical Data Systems (RNTDS)
  - 77% commonality across programs
  - 26% fewer labor hours (Good 1992)
Reported Benefits ... Continued

- Raytheon Missile Systems Division
  - 60% reuse
  - 50% productivity improvement (CARDS 1993)

- Magnavox Advanced Field Artillery Tactical Data Systems (AFATADS)
  - 68% gain in productivity (Payton 1992)
Reported Benefits ... Continued

• Fujitsu Software Development for Electronic Switching Systems (SDESS)
  - Before, 20% of ESSs delivered on schedule
  - After, 70% of ESSs delivered on schedule
  (Prieto 1991)
Reported Benefits ... Continued

- **Nippon Electronics Corporation (NEC) Software Engineering Laboratory**
  - Integrated reuse library into process and products of life cycle support environments
  - 6.7 fold productivity improvement
  - 2.8 fold quality improvement (CARDS 1993)
Appealing Opportunity

- Define a Level 3, Capability Maturity Model (CMM), process for domain and application engineering
  - Achieve ISO 9001 certification along the way

Based upon extrapolations from a tentative analysis of a number of reports, a plan and commitment of needed resources to assess, define, implement and continuously improve a life cycle process has a real potential to generate the following benefits for about $1375 per software engineer over about 3.5 years:

- Productivity gain per year 35%
- Early detection gain (i.e., pre-test) per year 22%
- Yearly reduction in time-to-market 19%
- Yearly reduction in post-release defect reports 39%
- Business value return-on-investment 5/1
Appealing Opportunity Continued

The report by Herbsleb et al (1994) is based upon empirical studies of 13 participating organizations with formal programs of software process improvement. An extract of the summary of overall results follows (1994 Copyright by Carnegie Mellon University):
## Appealing Opportunity Continued

<table>
<thead>
<tr>
<th>CATEGORY</th>
<th>RANGE</th>
<th>MEDIAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Years in improvement program</td>
<td>1..9</td>
<td>3.5</td>
</tr>
<tr>
<td>Cost per software engineer</td>
<td>$490..$2004</td>
<td>$1375</td>
</tr>
<tr>
<td>Productivity gain per year</td>
<td>9%..67%</td>
<td>35%</td>
</tr>
<tr>
<td>Early detection gain per year</td>
<td>6%..25%</td>
<td>22%</td>
</tr>
<tr>
<td>Yearly reduction in time to market</td>
<td>15%..23%</td>
<td>19%</td>
</tr>
<tr>
<td>Yearly reduction in post-release defect reports</td>
<td>10%..94%</td>
<td>39%</td>
</tr>
<tr>
<td>Business value of ROI</td>
<td>4.0..8.8</td>
<td>5.0</td>
</tr>
</tbody>
</table>
Appealing Opportunity Continued

- Note that domain and application engineering processes, the object paradigm, system engineering, etc. were not prerequisites for these programs
  - e.g., traditional methodologies for point solution, software engineering were acceptable for inclusion in these processes
Some Relevant Definitions and Types of Models
Some Relevant Definitions

Life Cycle: conception to retirement

Domain: problem space for a family of applications with similar requirements

Domain Engineering:
Analysis, design, implementation, and maintenance and evolution of domain models and Domain Specific Software Architectures.
Some Relevant Definitions Continued

A cornerstone of the concept of: *Process-Driven, Domain-Specific, Architecture-Centric, Library-Based Approaches* to Life Cycle Reuse of Processes and Products.
Some Relevant Definitions Continued

Domain Specific Software Architecture (DSSA): the structure and relationships among the software segments of a generic solution architecture (GA) for a product line within a domain. Such DSSAs (also called GAs) are the basis for reuse-in-the-large for future applications of the product line within the domain.
Some Relevant Definitions Continued

Product Line: solution to some distinct subdomain of an application problem space

Products: instances of a (part of a) product line
Some Relevant Definitions Continued

Application Engineering:
Analysis, design, implementation, and maintenance and evolution of instances of products and product lines within a domain. Leverages DSSAs.
Some Relevant Definitions Continued

Process: the way an organization performs and manages its work

“Process: -- The logical organization of people, machines, tools, methods, and procedures into work activities designed to produce a specified end result (work product).” (DoD 1991)
Some Relevant Definitions Continued

Risk: Combination of

• what can go wrong

• what is the likelihood of a particular “something” going wrong

• what are the possible consequences if it does go wrong (Charette, 1989)
Some Relevant Definitions Continued

Risk management principles should be used to determine the ‘life cycle model(s)’ and ‘process model(s)’ to be used by an organization on a project in a given domain.
Some Relevant Definitions Continued

Models: a representation at one or more levels of abstraction of something of interest; e.g., processes, products, interfaces

Life Cycle Models: represents the roles, products and interfaces across the life cycle for an organization’s work in a given domain; e.g., the Clear Lake Life Cycle Model which contains four levels of abstraction
Some Relevant Definitions Continued

Process Models: process definitions for the roles of a life cycle model; e.g., waterfall, spiral, incremental evolution
An Example of the Clear Lake Life Cycle Model
Phases of this Life Cycle Model

(McKay, Atkinson et al, 1992)
P(0)  Concept Exploration
P(1)  System Requirements Definition
P(2)  Software, Hardware and Communications, and Human Factors Requirements Definitions
  - Note that these three “threads” run through the rest of the life cycle.
P(3)  Architectural Design
P(4)  Detailed Design
Phases of this Life Cycle Model Continued

P(5) Unit Development and Testing
P(6) Integration
   - Acceptance Testing occurs after this phase
P(7) Maintenance and Operation
Some Relevant Definitions Continued

Activities: series of actions or tasks; may take place within a phase or across phases

Phases: has a defined set of inputs and contexts to trigger an iteration

• through the phase, and has a defined set of outputs and effects for each iteration
Some Relevant Definitions Continued

Baseline: for any phase of the life cycle, a set of artifacts that have been reviewed and approved for configuration management in accord with policies for the project
Some Attributes of Engineering Processes and Products

<table>
<thead>
<tr>
<th></th>
<th>Tech</th>
<th>Mgt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Products</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* Some Common Cell Attributes
  - Risks
  - Models
  - Standards
  - Metrics
  - Education & Training
  - Policies
Four Engineering Roles and Products

- Dev & Main
- Quality & Safety Mgt
- Products
- Project Mgt.
- Life Cycle Support Envir.
  (Infrastructure Team)
Three Engineering Environments for Computer Systems and Applications

- Host Environment
  - Develop and Sustain

- Integration Environment
  - Integrate and Monitor / Control

- Target Environment
  - Deploy and Operate
Process Models: Three Approaches

- Waterfall Model:
  - Acceptance Testing
  - P(0) P(1) P(2)

- Spiral Model:
  - Acceptance Testing
  - Risk Driven Spiral Model

- Spiral Model with Incremental Evolution and Risk Mitigation:
  - Acceptance Testing
  - 1 2 3 4 5 6
Process Models: Three Approaches Continued

Waterfall Deliverable (developed as one piece)

Spiral Deliverables (by risk ordered segments)

Incremental Evolution (by increments of risk ordered segments)
A Sample Form for Addressing
Issues in Risk Analysis and Management

Background

Risks can be discovered, created and/or exacerbated at any point in the life cycle of a complex computing system. Many important risks cannot be avoided in such systems. The engineering challenge is to improve our capabilities for risk analysis and management (Charette, 1989). Based upon research by Boehm and Charette, ten, prioritized risk items are identified below
A Sample Form for Addressing Issues in Risk Analysis and Management

along with suggested weights and guidelines for applying risk assessments to the segments of a project at each iteration through each phase.
A Sample Form .. Continued

A Prioritized, Top-Ten List of Software Risk Items (Items 1..10 by Boehm, 1988, Bullets and weights added by McKay, 1993, for use in the ROSE Project)

1. Personnel Shortfalls
   • Sw engineering expertise   (Max. Weight = 35)
   • Domain expertise           (Max. Weight = 25)
   • Process experience         (Max. Weight = 15)
A Sample Form .. Continued

2. Unrealistic plans (Third bullet added by McKay)
   • Schedule (Max. Weight = 20)
   • Budget (Max. Weight = 10)
   • Automated support (Max. Weight = 8)

3. Developing wrong functions
   (Max. Weight = 7)

4. Developing the wrong user interface
   (Max. Weight = 6)

5. Gold plating
   (Max. Weight = 4)
A Sample Form .. Continued

6. Continuous requirements changes
   (Max. Weight = 4)
   (McKay: Careful! This can change the other risks!)

7. Externally furnished components
   (Max. Weight = 4)

8. External tasks
   (Max. Weight = 4)

9. Performance shortfalls
   (Max. Weight = 4)

10. Straining the state-of-the-art
A Sample Form .. Continued

Guidelines:

1. The above weights can and should be tailored with time and experience to particular organizations and teams working in specific product lines and domains. However, the relative positions of the top two risk items compared to the remaining eight are very likely to remain the same. That is, research currently indicates that either of these top two items outweighs the
A Sample Form .. Continued

remaining eight as a predictor of project success provided item 6 is kept under control.

2. Each organization and team should develop guidelines as needed to further refine the assignment of the suggested weights. An assignment of "0" means little-or-no risk is believed to be associated with this item. An assignment of the maximum weight means maximum risk is believed to be associated with the item.
3. Upon exceeding some predetermined threshold for total risk assessed in a segment (or for an important part of a segment), a risk mitigation plan should be developed, executed, monitored and refined for the segment (or part). Such risk assessment should be applied to each iteration of each segment through a phase. A suggested initial threshold for the total risk score that would trigger this action is 30 points.
A Sample Form .. Continued

4. Upon exceeding some higher, predetermined threshold (suggested as 60 points initially) for total risk assessed in a segment (or for an important part of a segment) within an iteration of a phase, a risk mitigation plan should include at least two distinct sets of activities where either or both may result in mitigating the risk(s) satisfactorily.
Contrasting Scenarios
Contrasting Scenarios

Traditional “Point Solution” Based Approaches vs. “Domain Engineering” Based Approaches

• Scenario
  - The client proposes a new application, product or product line within a domain
    - Point Solution Approach
    - Domain Engineering Approach
Contrasting Scenarios Continued

• Point Solution Approach
  - for this specific project (i.e., largely independent of other applications in this domain), determine the Concept of Operations, System Requirements and Constraints, etc. and a Design and compose/generate accordingly
Contrasting Scenarios Continued

- Domain Engineering Approach
  - for this proposed member of the application domain, determine the commonalities and differences with the domain’s Concept of Operations, System Requirements and Constraints, .. GAs, Implementation Models, etc. and compose/generate accordingly
Introduction to
Example Problem 1
Example Problem 1

Assume you have been assigned the responsibility for leading a team to develop two new product lines in the home security application domain: the home/business computer monitoring system (HBCMS) and the community services coordination computing system (CSCCS). Assume the business case has been made for each product line and the scoping activities have resulted in acceptance of the details provided in the following paragraphs. You have been assured by marketing that if you can develop a highly reliable system for a reasonable cost for residents and business owners of “rural and urban communities”, there is a large market and a large potential return-on-investment. (For purposes of this project, ‘communities’ are defined as a logical collection of homeowners or business owners being served by a common set of: a single police department, a single fire department, a single emergency health-care system and a single cellular phone system.) External computing systems your product lines must interact with over the cellular phone system include: the police department computing system, the fire department computing system, and the ambulance/paramedic computing system.
Example Problem 1 Continued

The HBCMS consists of a local computing system (LCS) which monitors (and can report) ‘through-the-air’ signals from on-site sensor systems. Each sensor system will periodically respond to a ‘through-the-air’ command from the LCS to report its identity and status. The status will include its function (e.g., door is closed or open) and its power (e.g., battery is/is-not in the last 15% of an acceptable operational life). There are currently six classes of sensor systems defined in the software, but more are possible. The number of instances of each sensor class for a given site is software defined in the site configuration file. The six classes are:

- door/window: armed (closed, open), disarmed
- window pane: armed (intact, broken), disarmed
- motion detector: armed (no motion, motion), disarmed
- personal threat: armed (no threat, threat), disarmed
- health alarm: armed (no alarm, alarm), disarmed
- smoke detector: armed (no alarm, alarm), disarmed.

The HBCMS sensors can be set by the user (via the local computing system)
Example Problem 1 Continued

to one of three modes: Disarmed (all classes), Normal (motion detector and door/window disarmed, all others armed), Armed (all classes). If the alarm is not from a personal threat sensor, then once the LCS detects an alert from an armed sensor, the LCS sounds an alarm and activates a timer. The time-out value of the timer was established by the owner in the site configuration file. If the time limit expires, the LCS notifies the CSCCS which immediately calls the voice phone number of the site reporting the alarm. If a human answers within six rings and if the human can provide the previously stored password and assure the operations personnel at the CSCCS that all is well, then the CSCCS will disregard this particular report (but not subsequent reports). If no human satisfies these conditions within six rings, the CSCCS will notify the appropriate authorities for the type of alarm (e.g., police, fire, ambulance), location and time date stamp. Fire alarms are sent electronically to the fire department along with drawings of the building, placement of the smoke alarms and maps of how to get to the alarm site from key community locations. Medical alarms are sent electronically to the ambulance services along with vital medical histories of residents and maps of how to get to the alarm site from key community locations. Police alarms are sent electronically to the police department along with drawings of the building and the current alarm locations and with maps of how to get to the
Example Problem 1 Continued

alarm site from key community locations. If the alarm is from a personal threat sensor, the LCS sends an electronic emergency message to the police department. Next, the LCS will report this action to the CSCCS which will then send to the police station maps of how to get to the alarm site from key community locations plus any information regarding past threats or problems at this site. If the LCS ever wants to report any alarm to the CSCCS and it is not responsive by the third try, the LCS will report the alarm directly to the appropriate department identifying the type of alarm, the time date stamp, and the address of the alarm site. No maps, medical histories, or sensor locations are provided by the LCS during such reports. In addition to issuing “when observed” alarms to the CSCCS, the LCS will issue periodic health and status reports and will also respond to queries from the CSCCS to obtain such information.

In addition to the functionality for the CSCCS that can be inferred from the preceding discussion of the HBCMS, the CSCCS also maintains historical records and trends for each individual HBCMS site, for the collective HBCMS sites and for associated transactions. Normally, the CSCCS depends upon each HBCMS to report in by predetermined intervals (i.e., the CSCCS does not operate in polling mode). However, the CSCCS can query any HBCMS
Example Problem 1 Continued

that has missed two reporting cycles. Failure of a remote site to respond correctly will cause the CSCCS to notify the police department of the problem and send them a map to the non-responsive site.

Note that this problem is intended to be fairly realistic including the incomplete and ambiguous narrative problem description. Record any assumptions you have to make to evolve your solutions.

Assignment:

1. a) Prepare a concept of operations model for each product line (i.e., produce a complete Abstract Interface Specification -- AIS -- for each product line and an object relationship model depicting key relationships in the structure of the system).
   b) Prepare a behavior model for each product line (i.e., a state diagram).
   c) Prepare an interaction model for each product line (i.e., message passing diagrams).
   d) Initiate a domain dictionary that covers the issues identified in a) c).
Example Problem 1 Continued

2.a) Produce an object model of the systems level requirements and constraints for the two product lines. The model should leverage reuse potential (i.e., a combined model with a finer granularity of detail which is organized to maximize commonalities).
   b) Prepare a behavior model for each object class in the requirements model (i.e., a state diagram).
   c) Prepare an interaction model for each object class in the requirements model (i.e., message passing diagrams).
   d) Update the domain dictionary.

3.a) Prepare a Change Request (CR) so that, after an appropriate external computing system has been notified of an alarm, the pager of an owner can also be notified.
   b) Do an impact analysis on the first two phases.

4.a) Increasingly, communities are installing traffic control computing systems that regulate the flow of traffic during peak and emergency conditions by controlling the community traffic lights (e.g., make sure the fire trucks on the way to an emergency have all green lights). Prepare a CR that will allow your product lines to take advantage of these traffic control
Example Problem 1 Continued

services.
   b) Do an impact analysis on the first two phases.

5.a) Unfortunately, because of a number of inferior products from competitors, many local communities have now banned HBCMSs from reporting directly to the community computing systems. Instead, all alarm reports must now come from the centralized CSCCS of licensed operating companies. Prepare a Discrepancy Report to enable the required modifications as quickly as they can be safely and affordable made.
   b) Do an impact analysis on the first two phases.
First Pass at a WBS for Incremental Evolution

Work Breakdown Structure:

Pass(1)

Developmental Increments:

DI(1) => High reuse ‘have-to-haves’

DI(2) => Lower reuse ‘have-to-haves’ + higher reuse ‘nice-to-haves’

DI(3) => Complete through acceptance testing

1 => HBCMS.Sensors
2 => HBCMS.LCS.Admin
3 => HBCMS.LCS.Common
4 => HBCMS.LCS.Personal
5 => CSCCS.Normal
6 => CSCCS.Admin
Key Features of the Object Paradigm Used in This Approach
Object Classes and Relationship Sets/
Objects and Relationships
Some Common Object Concepts

- Note that this section *reviews* some ‘common’ concepts of object paradigms and describes some additional concepts to support robustness, concurrency and distribution in mission and safety critical (MASC) applications (McKay and Atkinson, 1995, 1992)

- objects are instances of object classes
- objects possess state, behavior and relationships
- objects support abstraction, encapsulation, information hiding, inheritance, polymorphism and
Robust, Concurrent Objects

Concurrency =>
• multiple threads of control
  - may be encapsulated within an object
  - may exist among objects of the system
    - distributed or local objects
Robust, Concurrent Objects Continued

Robust =>
- a thread of control may be transformed from ‘normal’ execution to ‘exception handling’ execution when the occurrence of an exception is recognized
  - exceptions may be user defined or predefined
  - handlers may be used to avoid or to tolerate predetermined classes of faults and failures
Some Additional Object Concepts Used in This Approach

Object Classes may be used to define either ‘active’ or ‘passive’ objects

- **Active** => instance of an abstract process type
  - i.e., possesses its own, externally visible, thread of control (e.g., an Ada task)
- **Passive** => instance of an abstract data type
  - i.e., must borrow a thread of control (via a method call) since it has no externally visible thread of its own
Some Additional Object Concepts Continued

Objects may be conceptually separated into an implementation part and an Abstract Interface Specification (AIS) part
• the AIS offers a ‘black box’ view of relationships, functionality, states, behavior and constraints
Some Additional Object Concepts Continued

Abstract Interface Specification

• Context
  - Relationships/ Constraints/ Exceptions/ ..

• Provided Interface
  - Resources/ Constraints/ Exceptions/ ..
  - Services/ Constraints/ Exceptions/..

• Required Interface
  - Resources/ Constraints/ Exceptions/..
  - Services/ Constraints/ Exceptions/..

• States
  - Values/ Constraints/ Exceptions/..
  - Sequences/ Constraints/ Exceptions/..
Some Additional Object Concepts Continued

Relationship Sets define the types of associations between and among Object Classes
• ‘Relationship Sets’ are as essential to well-engineered object models as ‘Object Classes’. Therefore, they are both ‘first class’ modeling constructs
  - e.g., both can be ‘attributed’, their attributes can be associated with constraints and exceptions, etc
Some Additional Object Concepts Continued

- Relationships are instances of Relationship Sets
- There are six types of Relationship Sets
- Objects can only interact via a message passing relationship
Some Additional Object Concepts Continued

Types of Relationship Sets

- Generalization/ (Is specialization of)
- Aggregation/ (Is part of)
- Association/ (Is member of)
- Parameterization/ (Is from generic template of)
- Instantiation/ (Is an instance of)
- Interaction/ (Is message participant with)
Some Additional Object Concepts Continued

Interaction/ (Is message participant with)
- Message Id
- Message Type
- Precondition
- Postcondition
- Source Id
- Destination Id
Some Additional Object Concepts Continued

• Service Id
  - Resource List
  - Service Type
    - Asynchronous (non-blocking)
    - Synchronous (blocking)
      - Remote Procedure Call
      - Rendezvous
Object Models, Views, and Product Sets
Object Models, Views and Product Sets

For any iteration of a process model in any phase of a life cycle model, the deliverables to be evolved for baselining include:

• a set of products with five interrelated parts

• a risk management view based upon the knowledge and insights acquired during this process iteration
Object Models, Views and Product Sets Continued

• a project planning, assessment and control view based upon the knowledge and insights acquired during this process iteration of evolving the product set and the risk management view
Object Models, Views and Product Sets Continued

A Set of Products with Five Interrelated Parts
• a set of AIS’s (one for each object class)
• an Object Relationship Model (ORM) depicting the object classes and relationship sets for the system
• a set of Object Behavior Models (OBM’s) (one for each object class) depicting states, transitions, constraints and exceptions
Object Models, Views and Product Sets Continued

• a set of Object Interaction Diagrams (OIM’s) depicting message-based interactions between and among the associated object classes
• a domain dictionary reflecting a ‘natural language’ perspective of the domain and application object classes and relationship sets
Object Models, Views and Product Sets Continued

Views => either
• hierarchy of abstractions
• supplemental information accessible only to authorized view holders

Diagram:

A → D → E
B → E
C → E

Diagram:

view holders
1

view

public info

view of
other info

2
Object Models, Views and Product Sets Continued

ORM for $P(O)$. 

Environment

(is composed of)

External Human Roles

External Computing Systems

Domains

External Property

Subdomain (1)/Product Line (1)

... 

Subdomain (n)/Product Line (n)
Object Models, Views and Product Sets Continued

Subdomain (n)/
Product Line (n)

Environment => Physical environment (e.g., ‘rugged’, ‘laboratory’)

External => Interface design controlled by group external to the project development team

Internal => Interface design controlled by the project development team
Object Models, Views and Product Sets Continued

Use Case Description (Jacobson et al, 1992)
• partial AIS
  - for any object class of interest (such as a product line for a subdomain), a partial AIS of an external or other internal object class describes only the parts of the partial AIS which are directly involved in a message passing relationship with the object class of interest
Object Models, Views and Product Sets Continued

- identifies
  - actors (who interact with the system)
  - preconditions
  - description
  - exceptions
  - postconditions
Outline of Transformation
Team’s Analysis Steps
Outline of Transformation Team’s Analysis

• P(0): Concept Exploration in the Domain
  - Business Models
    - identify product lines, their market potential, risks & opportunities, etc
  - Scoping Models
    - identify what part of the promising business models can be accomplished with available resources and appropriate risk management
  - Concept of Operations for Selected Product Lines
• P(1): Systems Requirements and Constraints for Selected Product Lines in the Domain
• P(2): Software/ Hardware and Communications/ Human Interfaces Requirements and Constraints for Selected Product Lines in the Domain
Example Problem 1
Security Domain and Product Lines

ORM for P(O)...

Environment

Home

Central

External Human Roles

External Computing Systems

Domains

External Property

Maps

Address & Sensor Locs

Phone Sys

HBCMS

CSCCS

Two Product Lines

Police

Health

Fire
Partial AIS for HBCMS, DI(1)

Context
• External: Police, Fire, Health
• Internal: CSCCS

Provided Interface
• procedure Ad_Hoc_Query ()

Required Interface
• procedure Ad_Hoc_Query_Response ()
• procedure Scheduled_Report ()
• procedure Alarm_Report ()
  - stub ‘procedure Add_or_Delete_HBCMS ()’
Partial AIS for HBCMS, DI(1) Continued

State Constraints
- Values: Initial, Monitoring (normal report, ad hoc report, alarm report)
- Sequences: (see OBM)
P(0) HBCMS Object Behavior Model: Pass(1)

HBCMS OBM, Provided I/F View

Monitoring

Normal Report

[@ scheduled report time]

Ad Hoc Report

[@ query received]

Alarm Report

[@ sensor alarm]
P(0) HBCMS Object Interaction Model: Pass(1)

CSCCS

HBCMS OIM

Messages

pre

normal

ad hoc

alarm
Software System, HBCMS, DI (1), P (0)
Introduction to
Example Problem 2
Example Problem 2

You have just been hired by a manufacturer of television sets and other commercial electronics products as a computer systems engineer to lead a team of three people to specify the design architecture (i.e., P(3) process and products) for a new class of remote T.V. tuners and a new class of local T.V. tuners. The local and remote tuners are to be functionally identical with two currently identified exceptions. First, if competitive signals arrive simultaneously from both the local and the remote tuners, the local signals have precedence. Second, if the battery in the remote tuner drops below nine volts at 100 milliAmps, no specifications guarantee the effects of trying to use the remote tuner except that the precedence relationship of the first exception cannot be violated.

Unfortunately, the P(2) product set is less complete and confidence-inspiring than you would prefer, but you are responsible for leading your team from this point forward. Producing an improved P(2) product set is not an option for your team at this time. Therefore, your team must use the information described below plus your own collective energies and domain knowledge to evolve the P(3) product set as soon as possible. You have educated and motivated team members but this is their first time to work with each other or with you or the Clear Lake Life Cycle Model you intend to use to model,
Example Problem 2 Continued

monitor and control your process. You have decided to assign one team member to act as your “Transform Team” for all processes, products and parts of P(3). Similarly, one member is assigned to all P(3) Quality and Safety Management and one to all P(3) Project and Configuration Management. Later, you will rotate assignments as part of your longer term team building goals. You, of course, are responsible for the system at the P(3) phase and have appropriate authority to lead your team.

The local tuner is installed inside a certain class of T.V. sets and powered by a local Alternating Current (A.C.) wall plug in the user’s environment (U.S., European or Japanese standards for household A.C. power). The remote tuner is hand held and battery powered. Externally visible to users is a three digit, on-screen display for the selected channel (range 000 to 999) plus a channel “picture”, speaker sound, and an on-screen bar graph for volume (range 0% to 100%). This channel and volume, on-screen display is visible for ten seconds after the latest change from either the local or remote tuners. The channel and volume settings, once set, are retained until they are changed or power is removed. (Turning on the set always causes channel X and Y% volume to be selected where X and Y are set to the user’s choice when the set is “initialized” for use by the user. Initialization can be
Example Problem 2 Continued

repeated by the user as desired.) Volume can be increased or decreased in 5% increments/decrements by depressing and releasing the logical equivalents of designated “push buttons”. Channels can also be incremented or decremented via buttons according to user programmed identification of channels of interest. (Channels of interest are established by the user at “initialization time”.) For example, depressing the “increase” channel button three times after starting on channel 2 can cause the channel selector to progress from 2 to 8 to 11 to 13 in the Houston area. Alternatively for channel selection, a three digit channel can be identified by depressing three buttons in sequence from a set of ten buttons labeled 0..9. (Entering a non-existent channel produces a “picture” and “sound” of “noise/snow”. Entering anything other than a numeric digit at any time before the third digit is entered will cancel the preceding digits and effect any legal response to what was entered. This includes turning the power off.)

With a fresh battery, the remote tuner can operate the T.V. from within a 45 foot radius enclosing a 60 degree arc from either side of a center line perpendicular to the front of the T.V. set. As does the local tuner adjustments, the remote tuner has controls for: on/off, volume up/down, channel up/down and three digit, direct channel selection. The remote tuner
Example Problem 2 Continued

type is to use the company’s “standard” general purpose, embedded target environment: microprocessor type, Random Access Memory (RAM) type, bus type, Programmable Read Only Memory (PROM) type, transmitter type, antennae type, battery type, user input switch types, and software tool sets and standards. The local tuner type also is to use the above company standards except that a receiver type is needed instead of a transmitter type and an A.C. power supply type is to be used instead of a battery type.

Your second team meeting identifies three “physical object classes” and one “logical object class” for your P(3) design architecture. The three physical object classes specify instances of the: local T.V. tuner and set, the remote tuner and the user. The last object class is active and the first two are passive. The logical object class is neutral containing type definitions and declarations common to the first two physical object classes. Your team assignment is to evolve a first pass at the P(3) product set by the next meeting date. Some parts of the design specification are expected to remain “To Be Determined” (TBD) but the team is expected to:
• do as much as time and thought permit before the next meeting and
• identify any assumptions or TBD-answers to questions that are critical to your approach.
Summary
Summary

• Overview of Domain Engineering Concepts
  - Contrast with Point Solution Approach
  - Engineering Perspectives
  - Some Anticipated and Reported Benefits
  - Some Relevant Definitions and Types of Models
Summary Continued

• Key Features of the Object Paradigm Used in This Approach
  - Object Classes and Relationship Sets
  - Object Models, Views & Product Sets
  - Outline of Transformation Team’s Analysis
  - Example Problems
References
References


References Continued


References Continued


References Continued


References Continued


References Continued

Acknowledgements
Acknowledgements

This research has been partially supported by three NASA-sponsored projects:
• RBSE
• MISSION
• Joint JSC/UHCL APSE Beta Test Site

The author wishes to thank:
• the NASA sponsors and monitors
• fellow researchers from the faculty and students at the university
• fellow researchers from industry
Acknowledgements Continued

• the support staff from RICIS and NAS at UHCL
• the volunteers on our Industrial Advisory Committee
Acknowledgements Continued

In addition, the author wishes to thank the UHCL students of the Spring 1995 class of “SWEN 5233: Software Generation and Maintenance” for their invaluable help in evolving ‘Example Problem 1’ of this tutorial. The students were:

- Todd Dunnavant
- Jose Figueroa
- Annette Garrett
- Sudershan Jambulapati
- Shuihua Jiang
Acknowledgements Continued

• Lance Le
• Tam Pham
• Mongwei Hu-Popp
• I-Chung Liang
Acknowledgements Continued

• Fadi Riman
• Reed Sellers
• Anand Srinivasan
• Bill Van Zandt
• Mike Weisskopf
• Danny Wong