Goal of Presentation

- Understand how to perform Object-Oriented Design
  - Using
    - MBASE
    - Object-oriented techniques
    - RUP
    - Rational Rose
- Understand how to document design
Outline

- When Last We Met …
- Design Process Overview
- Design Process by Example
- Summary

MBASE Process Overview

Activities & Representative Amounts

- Process Activities
  - Requirements Capture
  - Analysis & Design
  - Implementation
  - Test

Supporting Activities

- Management
- Environment
- Deployment

Stages

<table>
<thead>
<tr>
<th>LCO</th>
<th>LCA</th>
<th>IOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception</td>
<td>Elaboration</td>
<td>Construction</td>
</tr>
</tbody>
</table>

Time

Iterations

- Iter. 1
- Iter. 2
- Iter. 3
- Iter. m
- Iter. m+1
Purposes of Architecture Analysis & Design

- To transform requirements into design of system
- To evolve robust architecture for system
- To adapt design to match implementation environment
  - Designing it for performance
    - RUP 2001
Object, Component, System, People, Organization, …. (cont.)

What You Need To Describe At Any Level?

- **Structure**
  - External Perspective (Specification/Interface)
  - Internal Perspective (Implementation)

- **Responsibilities**
  - its state
  - its dynamic behavior
  - the operations it can perform
  - the requests it makes of other objects
    - Each component/object collaborates (works) with other components/objects in performance of its responsibilities

- **Qualities**
  - Quantitative (e.g., size, speed)
  - Qualitative (e.g., “user friendly”, re-usable)
MBASE Describes **System Internal (Architectural) View** in SSAD Section 3

- **Structure**
- **Responsibilities**
  - its state
  - its dynamic behavior
  - the operations it can perform
  - the requests it makes of other objects
- **Qualities**
  - Quantitative
  - Qualitative

Define Architecture
Software Component Classifier Model For Full-text Title Database System, Top–Level

- **Decision**
  - Client-Server Model
Define Architecture
Software Component Classifier Model For Full-text Title Database System, Server Component

Note: using a style that allows nested component classifiers

Define Architecture
Deployment Diagram for Full-Text Title Database System
Define Analyze Classes
Enterprise Classification Model for Full-Text Title Database System

Analyzer Behavior (1st Build)
Search and Locate Journals
Outline

- When Last We Met…
- Design Process Overview
- Design Process by Example
- Summary

System Design

- Describe how system can be implemented in software
- Describes specific technology solutions that satisfy Project & System requirements
- 2-levels
  - High-level Architectural Design
    - Resolves Analysis issues
      - e.g. how will roles and states be handled, expand bi-directional relationships, break multi-way relationships, handle global and relational attributes, decompose Components into objects, complex dependencies & other constraint
  - Low-level Implementation Design
    - Implementation considerations
      - e.g. use of databases, web-servers, hardware, critical algorithms, sequence, significant events, GUI’s, etc.
Outline

- When Last We Met…
- Design Process Overview
- Design Process by Example
- Summary
MBASE Describes System Internal (Implementation) View in SSAD Section 4

- Structure
- Responsibilities
  - its state
  - its dynamic behavior
  - the operations it can perform
  - the requests it makes of other objects
- Qualities
  - Quantitative
  - Qualitative

- Sec 4.1 & 4.5
- Sec 4.2
- Sec 4.3
- Sec 4.4

Refine Architecture – LCA or IOC

- Purpose:
  - Incorporate implementation decisions into architecture for system
    - Based on experience gained from similar systems or in similar problem domains
  - Identify Components
  - Understand
    - Hardware execution environment
    - Allocation of components to hardware

- Inputs:
  - LCO Architecture
  - LCO Logical Class Model
  - LCO Object Model
  - LCO Interaction Model
  - Architectural Patterns
  - L.O.S Requirements
  - Evolution Requirements
Refine Architecture – LCO or IOC

Artifacts

- Component Model
  - Define Implementation Component Model
    - Determine whether components are implemented by COTS or not
    - Determine interfaces of components
    - Define implementation-specific classes, object, interactions
  - Refine component classes
    - e.g. select specific implementation stereotypes
    - Add implementation specific components
  - Identify dependencies between components

- System Topology
  - Updated as appropriate to reflect changes to component model

- System Deployment Model
  - Updated as appropriate to reflect changes to component model

Refine Architecture – LCO or IOC

Implementation Component Model

- Purpose:
  - Describe how the components will be implemented
  - Define their interfaces
  - Describe COTS products that will be used to implement components, and how they are configured
  - Identify development technologies to be used
    - including database tables, Java, XML/HTML, HTTP servers, API’s, class libraries, design patterns

- Inputs:
  - See revised component model

- Artifacts
  - Interface Class Diagram(s)
  - Implementation Class Model
  - Description of COTS configuration
    - See MBASE COTS Integration Supplement
Implementation Component Model For Full-text Title Database System

- Document in SSAD 4.1.3
  - Library_User Component
    - Will be implemented with web-browser that access pages generated by Java Server Pages
  - System_Administrator Component
    - Will be implemented with web-browser that access pages generated by Java Server Pages
  - Server Component
    - Server::Business Objects
    - Server::Database
      - Will use MySQL as DBMS
      - Will implement classes in Java
  
  (See MBASE Guide & MBASE COTS Integration Supplement for details of section)

Refine Architecture – LCA or IOC
Component Model For Full-text Title Database System

- Revise to show any changes due to implementation decisions
  - Use of COTS
    - Web Browser
    - Web Server
    - MySQL
  - Dependency on
    - Java Framework
    - Web Framework
Define Implementation Class Model – LCO or LCA
For Library_User Component

Define Implementation Class Model – LCO or LCA
For Library_User Component

Define implementation of logical classes

Define attributes, operations, and relations for implementation classes
Refine Object Model – LCO or LCA
For Library_User Component

- Define implementation objects that are used to build component
  - May not need if implementation class model is real close to logical class model

Define Component Interfaces – LCO or LCA
For Server Component

- Define interfaces for Component
  - Define operations available in each interface
  - Define dependencies on other classes & interfaces
  - Note: deferring complete AdministratorServices interface until next build
Refine Architecture – LCA or IOC
Component Model For **Server** Component

- Revise to show any changes due to implementation decisions
  - Interface named SQLServices supplied by Database component
  - Database component will be Java Bean
  - New BusinessObjects Component (Java Bean)
    - Decouple higher-level components from decisions about Database (e.g. SQL)
    - Realizes AdministratorServices & UserServices interfaces

Define Component Interfaces – LCO or LCA
For Server::BusinessObjects Component

- Define **interfaces** for Component
  - Define operations available in each **interface**
  - Define dependencies on other classes & **interfaces**

Notes:
- Deferring complete AdministratorServices **interface** to later build
- This diagram is shown for completeness
  - Information already resented in Component Diagram for Server
Refined Logical Class Model – LCA or IOC

Component Model For **Server** Component

- When add BusinessObjects Component, moved this diagram (and logical Object-Structure Diagram) from Database subcomponent to Server component
- Realized that this diagram represented classes in both BusinessObject & Database subcomponents
- Alternative is to duplicate diagram in each component

Define Implementation Class Model – LCA or IOC

For Server Component

- Define implementation of logical classes
- Note: deferring Vendor_Set & User_Set to later build
Define Implementation Class Model – LCA or IOC
For Server::BusinessObjects Component

- Define attributes, operations, and relations for implementation classes
- Note: deferring Vendor_Set & User_Set implementations to later build

Refine Object Model – LCA or IOC
For Server::BusinessObjects Component

- Define implementation objects that are used to build component
  - May not need if implementation class model is real close to logical class model
Define Component Interfaces – LCA or IOC
For Server::Database Component

Define interfaces for Component
- Define operations available in each interface
- Define dependencies on other classes & interfaces

---

Define Implementation Class Model – LCA or IOC
For Server::Database Component

Define attributes, operations, and relations for implementation classes
Define Implementation Class Model – LCA or IOC
For Server::Database Component, Relational Database Schema

- Define attributes, operations, and relations for implementation classes

Refine Object Model – LCA or IOC
For Server::Database Component

- Define implementation objects that are used to build component
  - May not need if implementation class model is real close to logical class model

Spring 2003
Refine Interaction Model – LCA or IOC
For Search for Journals Use-Case

Define how objects interact to achieve use-cases
May not need if implementation class model is real close to logical class model

Refine Interaction Model – LCA or IOC
For JournalSetManager.Search Operation

Define how objects interact to achieve use-cases
May not need if implementation class model is real close to logical class model
Refine Interaction Model – LCA or IOC
For DatabaseConnection.Execute Operation

Define how objects interact to achieve use-cases
May not need if implementation class model is real close to logical class model

Refine Deployment Model – LCA or IOC
Component Configuration for ...

/Administrator : Workstation
  : Browser
  : Web Server
(from : Unix S...

/User : Workstation
  : Browser
  : Web Server
(from : Unix S...

Define how software components are allocated to hardware nodes
May not need if Implementation Deployment Model is same to Logical Deployment Model
Refine Deployment Model – LCA or IOC
Component Configuration for : Unix Server

- Define how software components are allocated to hardware nodes
- May not need if Implementation Deployment Model is same to Logical Deployment Model

Define Artifact Configuration Model

- Purpose:
  - Define artifacts that will be produced during implementation
  - Source files
  - Scripts
  - Binary files
  - Dynamic or static link libraries
  - Database Tables
  - Define what components or classes are in each artifact?
  - Define which artifacts “know” each other

- Inputs:
  - Component Model
  - Implementation Class Model

- Artifacts:
  - Configuration Model

- Typically done at LCO & LCA only if creating executable prototype
Configuration Model – LCA or IOC
For Full-text Title Database System

- Define
  - File structure
  - What Components or classes are in each file
  - Directory structure

Outline

- When Last We Met…
- Design Process Overview
- Design Process by Example
- Summary
What You Need To Describe At Any Level?

- Structure
  - External Perspective (Specification/Interface)
  - Internal Perspective (Implementation)
- Responsibilities
  - its state
  - its dynamic behavior
  - the operations it can perform
  - the requests it makes of other objects
    - Each component/object collaborates (works) with other components/objects in performance of its responsibilities
- Qualities (Levels of Services)
  - Quantitative (e.g., size, speed)
  - Qualitative (e.g., “user friendly”, re-usable)

MBASE Describes System Internal (Architectural) View in SSAD Section 3

- Structure
- Responsibilities
  - its state
  - its dynamic behavior
  - the operations it can perform
  - the requests it makes of other objects
- Qualities
  - Quantitative
  - Qualitative

- Sec 3.1
- Sec 3.2
- Sec 3.3
- Sec 3.4
- Sec 3.5

Independent of implementation details
**MBASE Describes System Internal (Implementation) View in SSAD Section 4**

- **Structure**
- **Responsibilities**
  - its state
  - its dynamic behavior
  - the operations it can perform
  - the requests it makes of other objects
- **Qualities**
  - Quantitative
  - Qualitative

- Sec 4.1 & 4.5
- Sec 4.2
- Sec 4.3
- Sec 4.4

Independent of implementation details