Topics

- Package Design
  - Use Case Packages
  - Class Packages
- Component Design
- Deployment Design
- Collaboration Design
From analysis - Use Case Diagram

From analysis - Class Diagram
Package design

- Package - groups classes, use cases or other modeling elements
- Useful in large systems
- We distinguish between:
  - Use Case packages – emphasized in analysis
  - Class Packages – emphasized in design

Use case packages

Each package will eventually have more use cases than shown (in particular, «extend» and «include» use cases)
**Boundary class packages**

- Most classes that we defined in analysis represented **persistent database objects** ("business objects")
- **BCED application program classes** need to be considered as well

The functions of configuring computers and entering orders require Boundary Packages

- **<<boundary>> Configuration GUI**
- **<<boundary>> Order GUI**

**Entity class packages**

- **Persistent database classes** correspond to **Entity Classes in the application program**
- **Entity Packages** represent in-memory run-time structure for persistent database classes

- **<<entity>> Customers**
- **<<entity>> Computers**
- **<<entity>> Orders**

Includes classes Invoice and Payment
Control class packages

- **Control classes → Control Packages**
  - represent application logic
  - “glue” boundary and entity classes

```
<<control>>
Configure Process
```

```
<<control>>
Order Placement
```

DB interface class packages

- To mediate between entity classes and the database
- To handle connections, authorizations, transactions
- To hold “meta-information” about DB schema

```
<<db interface>>
CRUD
```

```
<<db interface>>
Connection
```

```
<<db interface>>
Schema
```
Component design

- **Components** – physical parts of the system
- **Component design** refers to the implementation platform for the system
- **OnLine Shopping** – Web application with database server

**Web application**
- “…Web system that allows its users to execute business logic with a web browser.”
- Business logic can reside on the server and/or on the client
- Client/Server system with a Web site

Implementing Web applications

- **Web pages**
  - Rendered in Internet client **browser**
  - Delivered by **Web server**
- **Web page document**
  - can be static (unmodifiable) or dynamic
  - can be a **form** that a user fills in
- **Frames**
  - divide the screen’s “real estate” so that the user can view multiple Web pages at the same time
- **Application server**
  - to manage the application logic
  - to monitor the application state
    - By storing **cookies** in the browser
  - Session timeouts
Implementing Web applications

- **Dynamic client pages**
  - **Script** — program interpreted by the browser
  - **Applet** — compiled component that executes in the browser’s context

- **Server pages** - Web pages with scripts executed by the server
  - Have access to DB server
  - Manage client sessions
  - Place cookies on the browser
  - Build client pages

---

Component diagram

- **Component**
  - Cohesive functional unit with clear interfaces
  - Replaceable part of the system
  - Can correspond to implementation of one or more Web pages
  - Can parallel Use Case Packages
Deployment design

- Assignment of objects to computing nodes

- Difficulties related to Web applications
  - Connectionless nature of Internet
  - Session management
    - Cookies
    - Distributed objects (CORBA, DCOM, EJB)
      - Application server between Web server and DB server
  - Web server as the routing point between all client browsers and the database
  - Security
  - Network loads, backups, etc.

Deploying Web applications

- Four tiers of computing nodes
  - Client with browser
    - Static and dynamic pages
    - Scripted pages and applets downloaded and run within the browser
  - Web server
    - Page requests from the browser
    - Generation of pages and code for execution on the client
  - Application server
    - Necessary with distributed objects
  - Database server
    - Data storage
    - Multi-user access
**Deployment Diagram**

- **Client Browser** → page requests → **Web Server**
- **Web Server** → database requests → **Database Server**

---

**Collaboration design**

- Architectural design ≡ packages, components
- Detailed design ≡ collaboration design
- Collaborations define the realization of
  - Use cases
  - Operations
- Collaboration design is conducted in parallel with the elaboration of
  - Use case models
  - Class models
  - most other models
Elaborating use cases

Use Case Specification: Order Configured Computer

1. UC15 Order Configured Computer

1.1 Brief Description

A customer fills in and submits a purchase order form. The system verifies the details and confirms or rejects the order.

2. Flow of Events

2.1 Basic Flow

2.1.1 UC15.1 The system displays the Order Entry form in the customer's Web browser. The form contains the following items:

- UC15.1.1 The title of the form is "Order Your Computer."
- UC15.1.2 Explanatory information is displayed below the title.

The text for explanatory information is:

"Please fill out the boxes in the form. Prompts for required items are in red color and in boldface. Press the Submit button to submit the form or Cancel button if you decide not to proceed with your order. You can cancel your order without penalty within 24 hours from the order's submission. You can cancel an order by using Web, email, fax, or phone."

The document "Use Case Specification: Update Order Status" describes how order can be..."
Structure of collaboration

- Structure of collaboration = collaboration
  Class Diagram extended with application program classes (BCED classes)
- Adheres to the enabling technology chosen for the application
- Difficulty:
  - The enabling technology may not be OO
- OnLine Shopping
  - Boundary classes – client pages, forms
  - Control classes – server pages

Using BCED approach

- Recommended practice - prefixing the class names with letters
  - b (Boundary), e.g. b_OrderClientPage
  - c (Control)
  - e (Entity)
  - d (Database Interface)
- Association and aggregation relationships to link BCED classes
- Instantiation relationships to signify messages that instantiate objects
  - User events leading to object instantiation can be named
**Boundary classes**

For use case “Order Configured Computer”

Association relationship

Instantiation relationship with named user event

**Control and entity classes**

For use case “Order Configured Computer”
Behavior of collaboration

- For use case “Order Configured Computer”
- Boundary and control objects

![Diagram of collaboration behavior]

Note from structural collaboration that b_OrderClientPage contains b_OrderClientForm

Behavior of collaboration

- For use case “Order Configured Computer”
- Entity and DB Interface objects

![Diagram of collaboration behavior]