CSC 330 OO Software Design

Discipline of Software Design
Objectives

- To explain what design is and how various types of design deal with different aspects of a product
- To present design as problem solving and outline the roles of abstraction and modeling in design
- To place design in the software life cycle
- To survey software engineering design methods
Topics

- Software products and software design
- Abstraction and modeling
- Varieties of design
- Software design in the life cycle
- Software engineering design methods
Importance of Software Design

- We live in a designed world.
- Design is economically important and effects our quality of life.
- The quality of software design has important consequences that software designers should be aware of and take seriously.
A *software product* is an entity comprised of one or more programs, data, and supporting materials and services that satisfies client needs and desires either as an independent artifact or as essential ingredient in some other artifact.
Software Design Defined

Software designers do what designers in other disciplines do, except they do it for software products.

**Software design** is the activity of specifying the mature and composition of software products that satisfy client needs and desire, subject to constraints.
Design as Problem Solving

An especially fruitful way to think about design is as problem solving.

Advantages

- Suggests partitioning information between problem and solution
- Emphasizes that there may be more than one good solution (design)
- Suggests techniques such as changing the problem, trial and error, brainstorming, etc.
Abstraction

Abstraction is an important problem-solving technique, especially in software design.

**Abstraction** is suppressing or ignoring some properties of objects, events, or situations in favor of others.
Importance of Abstraction

Problem simplification

- Abstracting allows us to focus on the most important aspects of a problem in (partially) solving it.

Structuring problem solving

- Top-down strategy: Solve an abstract version of the problem, then add details (refinement)
- Bottom-up strategy: Solve parts of a problem and connect them for a complete solution
Modeling

A model represents a target by having model parts corresponding to target parts, with relationships between model parts corresponding to relationships between target parts.
Modeling in Design

Modeling is used for the following purposes:

- Problem understanding
- Design creation
- Design investigation
- Documentation

Modeling work because models abstract details of the target.

Models can fail if important and relevant details are left out.
Static and Dynamic Models

A static model represents aspects of programs that do not change during program execution.

A dynamic model represents what happens during program execution.

Static model examples include class and object models.

Dynamic model examples include state diagrams and sequence diagrams.
**Product vs. Engineering Design**

*Product designers* are concerned with styling and aesthetics, function and usability, manufacturability and manageability.

- Industrial designers, (building) architects, interior designers, graphic designers, etc.

*Engineering designers* are concerned with technical mechanisms and workings.

- Structural, mechanical, chemical, and electrical engineers

Design teams often include both product and engineering designers.
Software Product Design

Software product design is the activity of specifying software product features, capabilities, and interfaces to satisfy client needs and desires.

Requires skills in user interface and interaction design, communications, industrial design, and marketing.
Software Engineering Design

Software engineering design is the activity of specifying programs and sub-systems, and their constituent parts and workings, to meet software product specifications.

Requires skills in programming, algorithms, data structures, software design principles, practices, processes, techniques, architectures, and patterns.
The waterfall model captures the logical, but not the temporal, relationships between software development activities.
Design Across the Life Cycle

Software Product Life Cycle

- Requirements Specification
- Design
- Implementation
- Testing
- Maintenance

Product Design

Engineering Design

Product Redesign and Engineering Redesign
“What” Versus “How”

- Traditional way to make the distinction between requirements and design activities
- Not adequate because
  - Many “what” specifications turn out to be design decisions
  - Many “how” specifications turn out to be client or customer needs or desires
- Distinguish requirements from design based on problem solving: requirements activity formulates a problem solved in design
Design Problems and Solutions

Software Design

Problem: Needs, Desires, Constraints

Solution: Features and Capabilities

Product Design

Design Features and Capabilities

Solution: Interactions

Engineering Design

Create High-Level Design

Solution: SRS

Create Low-Level Design

Solution: High-Level Design

Write Code

Solution: Low-Level Design

Solution: Design Document

Solution: Code
A software design method is an orderly procedure for generating a precise and complete software design solution that meets clients needs and constraints.
Design Method Components

- **Design Process**—A collection of related tasks that transforms a set of inputs into a set of outputs
- **Design Notations**—A symbolic representational system
- **Design Heuristics**—Rules providing guidance, but no guarantee, for achieving some end

Design methods also use design principles stating characteristics of design that make them better or worse.
Design Method Timeline

1971 Niklaus Wirth introduces stepwise refinement.

1974 Stevens, Myers, Constantine introduce structured design.

Late 1970s to early 1980s Structured analysis and design methods are dominant.

Late 1980s Object-oriented analysis and design methods rise to prominence.

1995 UML 0.8 is released.

2004 UML 2.0 is released.
Software Design
Processes and Management
Confusion arises around the term design.
This confusion is removed by adopting the following terminology.

**Analysis** is breaking down a design problem to understand it.

**Resolution** is solving a design problem.
A Problem-Solving Process

1. Understand the problem
2. Generate candidate solutions
3. Evaluate candidate solutions
4. Select the best solution(s)
5. Iterate if no solution is adequate
6. Ensure the solution is complete and well-documented, and deliver it
A Generic Design Process

Generic Design
need : Problem
design : Solution

need

Analyze the Problem

Problem Statement

Resolve the Problem

[problem misunderstood]

[else]

design
A Design Resolution Process

This diagram shows details of the resolution activity from the previous diagram.
Design Process Characteristics

- The best solutions are rarely the first solutions designers think of.
  - *Designers should generate many candidate solutions.*

- The design process is highly iterative.
  - *Designers must frequently reanalyze the problem and must generate and improve solutions many times.*
Architectural and Detailed Design

**Architectural design** is high-level software engineering design resolution.

**Detailed design** is low-level software engineering design resolution.
Operations versus Projects

**Operations** are standardized activities that occur continuously or at regular intervals.
- Payroll
- Hiring and performance evaluation
- Shipping and receiving

**Projects** are one-time efforts to achieve a particular current goal.
- Process improvement
- Business restructuring
- *New product introduction (including design)*
Project Management Activities

- **Planning**—Formulating a scheme for doing a project.
- **Organizing**—Structuring the organizational entities involved in a project and assigning them authority and responsibilities.
- **Staffing**—Filling the positions in an organizational structure and keeping them filled.
- **Tracking**—Observing the progress of work and adjusting work and plans accordingly.
- **Leading**—Directing and supporting people doing project work.
**Project Planning**

- **Estimation** is calculation of the approximate cost, effort, time or resources required to achieve some end.

- A **schedule** specifies the start and duration of work tasks.

- Tasks are allocated resources based on the schedule and estimates.

- **Risk analysis** is an orderly process of identifying, understanding, and assessing **risks** (any occurrence with negative consequences).

- Policies, procedures, tools, and techniques are specified to govern work.
Project Organization and Staffing

Organizational structures
- Project organization
- Functional organization
- Matrix organization

Team structures
- Hierarchical teams
- Democratic teams

Staffing
- Often the single most important factor in success is having good people to do the work.
Project Tracking

Projects may not go as planned for many reasons.

- Resource consumption is not as expected.
- Tasks do not take as long as expected.
- Policies, procedures, tools, or techniques cause problems.
- Something bad occurs (illness, budget cuts, equipment failures, etc.

When plans fail they must be adjusted.
Leading a Project

- Direction is needed to follow plans, use resources efficiently, etc.
- Directing people is not enough—people need inspiration, help, a congenial work environment, emotional support, etc.
Iterative Planning and Tracking

Good planning requires knowledge of tasks and their costs, risks, and other details not known until the project is under way—but this is not known when plans are made.

Iterative planning and tracking is making a rough base or initial project plan, and refining it at fixed periods during a project in light of tracking data and completed work products.
Design Project Management

- All five project management activities are needed to manage a design project.
- Iterative planning and tracking is the best approach to planning and tracking.
- The design project decomposition on the next slide is useful for planning, organization, staffing, and tracking.
- Design constitutes the largest activity in software development, so design can drive an entire development project.
## Design Project Decomposition

<table>
<thead>
<tr>
<th>Work Phase</th>
<th>Typical Work Products</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Product Design</strong></td>
<td><strong>Analysis: Design Problem</strong></td>
</tr>
<tr>
<td></td>
<td>- Statement of interested parties, product concept, project scope, markets, business</td>
</tr>
<tr>
<td></td>
<td>goals</td>
</tr>
<tr>
<td></td>
<td>- Models (of the problem)</td>
</tr>
<tr>
<td></td>
<td>- Prototypes (exploring the problem)</td>
</tr>
<tr>
<td></td>
<td><strong>Analysis: Detailed Needs</strong></td>
</tr>
<tr>
<td></td>
<td>- Client surveys, questionnaires, interview transcripts, etc.</td>
</tr>
<tr>
<td></td>
<td>- Problem domain description</td>
</tr>
<tr>
<td></td>
<td>- Lists of needs, stakeholders</td>
</tr>
<tr>
<td></td>
<td>- Models (of the problem)</td>
</tr>
<tr>
<td></td>
<td>- Prototypes (exploring needs)</td>
</tr>
<tr>
<td></td>
<td><strong>Resolution: Product Specification</strong></td>
</tr>
<tr>
<td></td>
<td>- Requirements specifications</td>
</tr>
<tr>
<td></td>
<td>- Models (of the product)</td>
</tr>
<tr>
<td></td>
<td>- Prototypes (demonstrating the product)</td>
</tr>
<tr>
<td><strong>Engineering Design</strong></td>
<td><strong>Analysis</strong></td>
</tr>
<tr>
<td></td>
<td>- Models (of the engineering problem)</td>
</tr>
<tr>
<td></td>
<td>- Prototypes (exploring the problem)</td>
</tr>
<tr>
<td></td>
<td><strong>Resolution: Architectural Design</strong></td>
</tr>
<tr>
<td></td>
<td>- Architectural design models</td>
</tr>
<tr>
<td></td>
<td>- Architectural design specifications</td>
</tr>
<tr>
<td></td>
<td>- Architectural prototypes</td>
</tr>
<tr>
<td></td>
<td><strong>Resolution: Detailed Design</strong></td>
</tr>
<tr>
<td></td>
<td>- Detailed design models</td>
</tr>
<tr>
<td></td>
<td>- Detailed design specifications</td>
</tr>
<tr>
<td></td>
<td>- Detailed design prototypes</td>
</tr>
</tbody>
</table>