Introduction to Software Design

CSE-474 Software Design & Architecture

Dr. Awais Majeed

February 7, 2011

Outline

Software Design I

- A software design is a meaningful engineering representation of some software product that is to be built.
  - The process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization" [TAY59]
- A design can be traced to the customer's requirements and can be assessed for quality against predefined criteria.
- It lays down the groundwork for achieving non-functional requirements (performance, maintainability, reusability, etc.)
- It takes target technology into account (e.g., kind of middleware, database design, etc.)
- From abstractions to systems

Software Design II

- Abstractions allow us to ignore implementation details of procedures and data structures
- For large systems, we need to abstract away even more detail
- We need to represent higher level abstractions

Viewpoints/Representations

- Viewpoints help in creating abstractions
- A viewpoint tells you which details you can ignore when forming an abstraction
- It defines which details are relevant and which are not
- Viewpoints can overlap
  - Some aspects of a design are common to several viewpoints

Design Represention

- Help us to see the big picture
  - Allow us to communicate our designs with others
    - customers, managers, other developers,
    - people with varying technical expertise
  - Allow us to measure various quality attributes
    - completeness, consistency, complexity

Software Design Representations

Structural

- Static properties

Functional

- Tasks performed by system

Behavioural

- Cause and effect, system behaviour

Data Modeling

- Data objects and their relationships

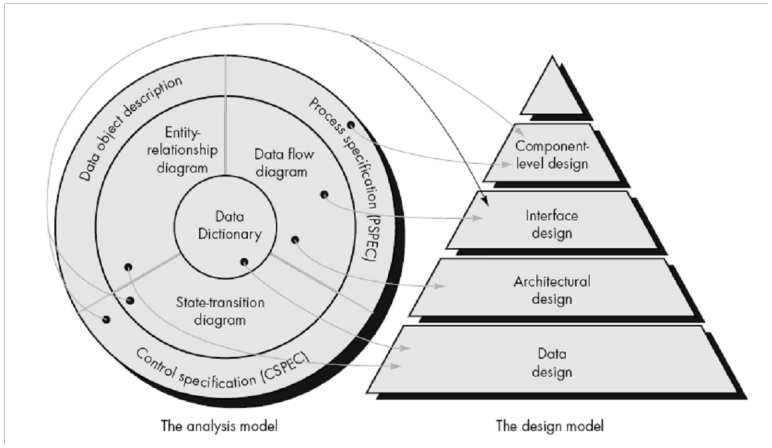
SDLC and Design

- In SDLC (Software Development Life Cycle), Design phase is one of the most important phases.
- In the software engineering context, design focuses on four major areas of concern: data, architecture, interfaces and procedures/components.

Design Process

- During the design process, the software specifications are transformed into design models that describe the details of the data structures, system architecture, interface, and components.
- The emphasis in design phase/process is on quality.
- This phase provides us with representation of software that can be assessed for quality.

Analysis to Software Design



Design Process activities

- Architectural design
- Abstract specification
- Interface design
- Component design
- Data structure design
- Algorithm design

Levels of Software Design

Architectural design (high-level design)

- Architecture - the overall structure, main modules and their connections
  - Addresses the main non-functional requirements (e.g., reliability, performance)
  - Hard to change
- Detailed design (low-level design)
  - The inner structure of the main modules
  - Detailed enough to be implemented in the programming language

Design Principles

Software Design Should be:

- Simple
- Correct & Complete
- Loosely coupled
- Understandable
- Adaptable

How to avoid bad design?

- Follow Design Principles
- Use established Design Patterns

Software Design Principles

- The design process should not suffer from tunnel vision
  - consider alternative approaches.
- The design should be traceable to the analysis model
- The design should not reinvent the wheel
- The design should minimise intellectual distance between the software and the problem as it exists in the real world.
- The design should exhibit uniformity and integration
  - A design is uniform if it appears that one person developed the whole thing.
  - A design is integrated if care is taken in defining interfaces between design components.
- The design should be structured to accommodate unusual circumstances, and if it must terminate processing, do so in a graceful manner.
- The design should be reviewed to minimize conceptual

Design Methods I

Software Design Methods

- Structured Methods
  - Process functions are identified
- Object-Oriented
  - Develop an object model of a system
- Data-Oriented
  - Entities are determined for each sub-system, then entity inter-relationships are examined to develop the additional entities needed to support the relationships.
- Component-based
  - Divide the system into components
- Formal Methods

Design Methods II

- Requirements and programs are translated into mathematical notation
-