Software Engineering

The Design of Software Architecture

Avinanta Tarigan

Gunadarma University

Design Engineering

- Principles & Concepts to deliver high quality products
- Bring together
  - Creativity
  - Customer Requirement
  - Technical Considerations
- Well-designed software:
  - Firmness: no bugs
  - Commodity: suitable for the intended purposes
  - Delight: pleasure to use one
- Designer must practice diversification and the convergence

Analysis vs Design

- Analysis:
  - required data
  - function
  - behaviour
- Design:
  - a representation / model of software
  - data structures
  - architecture
  - interfaces
  - components and modules
- Miracle of SE: analysis $\rightarrow$design, design $\rightarrow$code

Design Model

- Data / Class Design
  - Realizations of Class-Based elements
  - Class Diagram, Analysis Packages, CRC models, Collaboration Diagrams
- Architectural Design
  - Major structural elements, architecture styles & patterns to achieve requirements
  - Class based elements + Flow oriented (DFD, Control-Flow)
- Interface Design
  - Communication with external entities
  - Interoperability & User interface design
  - Scenario-based: Use-cases, Activity, etc + Flow-oriented
- Component-Level
  - Structural Elements to procedural descriptions

Quality Guidelines

- A design should exhibit :
  - recognizable styles and patterns, good design characteristic
  - can be implemented in evolutionary fashion
- Modular
- Distinct representation of Data, Architecture, Interfaces, Components
- Reduced Complexity
- Representated by recognizeable notations for communication

Quality Attributes FURPS

- Functionality
  - Features, capabilities, security
- Usability
  - Aesthetics, consistency, documentation
- Reliability
  - Frequency & severity of failure, Recoverability, Predictability
- Performance
  - Processing Speed, response time, resource consumption, throughput, efficiency
- Supportability
  - Extensibility, Adaptibility, Serviceablity, Maintainability

Design Concepts I

- Abstractions
  - "abstraction is one of the fundamental ways that we as human cope with complexity"
  - procedural abstractions: sequence of instructions
  - data abstractions: named collection of data
- Architecture
  - Structural Models
  - Framework Models
  - Dynamic Models
  - Process Models
  - Functional Models
- Patterns
  - Contain essence of proven solution to recuring problem
  - Can be reused, formalized as guidance

Design Concepts II

- Modularity
  - Monolithic vs Modular
  - Manageable
  - Optimal Costs
- Information Hiding
  - How to decompose software to obtain best st of modules
  - Parnas: principles of information hiding
- Functional Independence
  - Modularity
  - Qualitative Criteria:
    - Cohesion: information hiding
    - Coupling: interdependency
- Refinement
  - Successively refining levels of procedural details

Design Concepts III

- Process of Elaboration
- Refactoring
  - Reorganization to achieve simplified design
- Design Classes
  - Set of solutions
  - User interface, Business domain, Process, Persistens, System classes
  - Well-Formed Design Class:
    - Complete & Sufficient
    - Primitiveness
    - High Cohesion
    - Low Coupling

Design Model

- Data Design Elements
- Architectural Design Elements
- Interactive Design Elements
- Components-Level Design Elements
- Deployment-Level Design Elements

Design Pattern

- Repeated solutions / design
  - Producing Templates
    - Architectural, Aggregation, Idioms
  - Framework
    - Skeleton to achieve solution
    - Reusable components

Architectural Design

- Architecture $\longrightarrow$ Construction
- Stuctures of the system that comprise software components (visible properties & relationships among them)
  - Data Design
  - Architectural Styles & Patterns
  - Interactions among modules
  - Global properties of the assemblage
  - Integration of modules

Data Design

- Action that translates data objects in analysis model into data structures at the software component level
- At architectural level:
  - knowledge discovery in databases
  - the use of data warehouse
  - tools: SPSS, Business Objects, etc
- At component level:
  - representation of data structures
  - used directly by the developed software

Architectural Styles

- Data-centered Architecture
  - A data store provides data services to number of clients
  - Promotes Integrability
- Data-flow Architecture
  - Input is to be transformed through manipulation
  - Computation / manipulation is done by Filter
- Call-and-Return Architecture
  - Main-Program Sub-Program
  - Remote Procedure Call
- Object Oriented Architecture
- Layered Architecture

Patterns

- Concurrency
  - handling multiple-tasks at one time
  - the use of OS's native multitasking
  - task scheduler
- Persitence
  - Storing data in permanent location for later use
  - DBMS vs Application Level Persitence
- Distribution
  - Entities are working distributively
  - Way to connect
  - Nature of Communication (man-in-the-middle)

Architectural Design

- Representation in Context Level
  - Superordinate System
  - Subordinate System
  - Peer-level System
  - Actors
- Archetypes
  - Patterns that represents a core of abstraction
  - For example: server, client, middle-tier
  - Another example: node, detector, indicator, controller

Summary

- Holistic View of the system to be built
- Objective: Quality
- Translations from Analysis to Design, prepared for the Coding
- Styles and Patterns
- Data Design & Architectural Design