Software Engineering

End Semester Exam

Vasa Case Study

- Stakeholders involved
- How requirement change affected project
- Why communication between teams is important
- Importance of proper testing and reporting
- The importance of balancing functional and non-functional constraints
- A good software architect [7]

What is Architecture

- Software Architecture Definition
- Related terminologies Diagram
  [Reference Model, Architectural Pattern],
  [Reference Architecture],
  [Software Architecture]
- **Architectural Pattern** *Definition and need*
- Architectural Patterns [15]
- Pipe and Filter + *Diagram*
- **Reference Model**
- **Reference Architecture**
- Importance of Software Architecture [3]
- Consequences of Architectural Choice [7]
- How Architecture promotes reuse [4]

- **Module View** :
  - Decomposition
  - Uses
  - Layered
  - Class
- **Component and Connector** :
  - Client-Server

    - Concurrency
    - Process
    - Repository *(shared data)*
- **Allocation** :
    - Work Assignment
    - Deployment
    - Implementation

- Architectural Business Cycle
    - Stakeholders [5]
    - Architect’s Influences [4] + *Diagram*
    - Architect’s Activities [7]
    - Process Advice [7]
    - Good rules of thumb [8]

Documenting The Architecture

- Different Stakeholders have different needs -- require different views
- **Stakeholder View Table**
- Template :
    - Primary Presentation *Diagram*
    - Element Catalog
    - Context Diagram
    - Variability Guide
    - Architectural Background
    - Terms Used
    - Other information
- Documenting Interface [9]
- Architecture Description Language
    - Definition
    - Positives [5]
    - Negatives [4]
    - Examples

Designing The Architecture

- Architectural Patterns [15]
- **Layered**

    - Definition and understanding
    - Diagram
    - Implementation
- **Pipes and Filters**
    - Definition and understanding
    - Diagram and example
- **Blackboard**
    - Definition and understanding
    - Diagram
- **Broker**
    - Distributed systems -- ring and star, client-server
    - Applicability for system -- {system type}
    - Definition and understanding
    - Diagram

- Domain Specific Systems
- State Transition Systems

- **Microkernel**
    - Adaptable System -- applicability
    - Definition and understanding
    - Diagram

- Attribute Driven Systems
- **Attribute Driven Design Process**
    - Choose Module to Decompose
    - Refine Module
        - Choose **Architectural Drivers** [definition]
        - Choose **Architectural Pattern** [tactics]
        - Instantiate module
        - Allocate functionality
        - Represent using views -- one of each category
        - Define interfaces
        - Refine use cases and scenarios
    - Repeat till done

Quality Attributes and Tactics

- Qualities -- definition

- Tactics -- definition
- Quality Scenario
    - *Source*
    - *Stimulus*
    - *Artifact*
    - *Environment*
    - *Response*
    - *Response Measure*

- **Availability**
    - Table
    - Example Scenario
    - Tactics
        - Fault Detection
        - Fault Recovery
        - Fault Prevention

- **Performance**
    - Event Arrival Patterns [3]
    - Event Servicing
        - Latency
        - Jitter
        - Throughput
    - Table
    - Example Scenario
    - Tactics
        - Resource Demand
        - Resource Management
        - Resource Arbitration

- **Security**
    - Sub-criterions [6]
    - Table
    - Example Scenario
    - Tactics
        - Resisting Attacks
        - Detecting Attacks
        - Recovering from Attacks

- **Modifiability**
    - Table
    - Example Scenario
    - Tactics
        - Localization of Changes
        - Prevent Ripple Effect
        - Defer Binding Time

- **Testability**
    - Table
    - Example Scenario
    - Tactics
        - Manage IO
        - Internal Monitoring

- **Usability**
    - Definition and understanding
    - Table
    - Example Scenario
    - Tactics
        - Design Time
        - Runtime