Software System Design (Architecture)

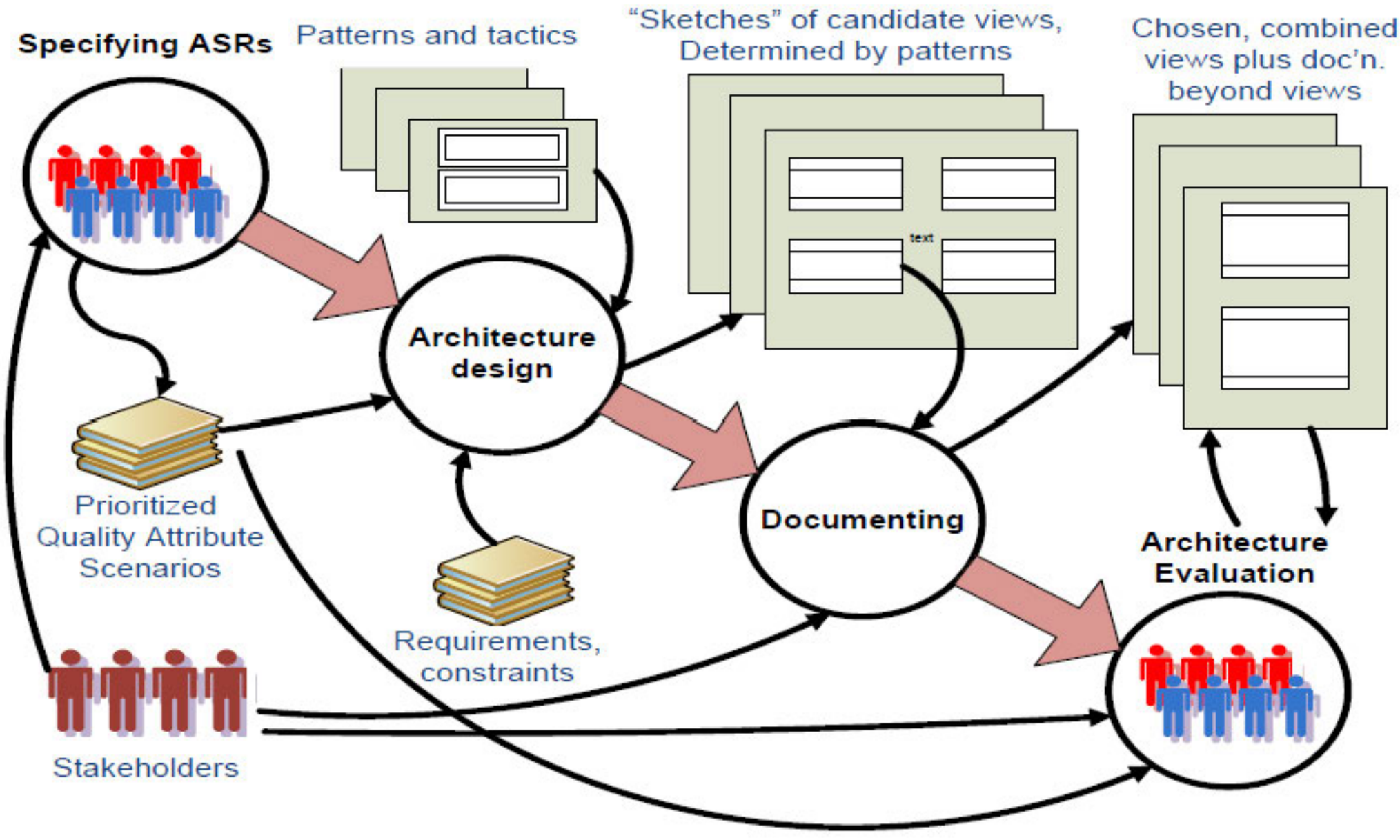
Outline

- Software Architecture
- Quality Attributes
- Architecture Patterns
- Designing Software Architecture
- Documenting Software Architecture
- Evaluating Software Architecture
- Describing Architecture
- Microservice Architecture*

Software Architecture in General

- What is software architecture?
  - Structure, Elements, Relationships, Design
- What does a software architect do?
- Where do architectures come from?
  - NFRs, ASRs, Quality Requirements; Stakeholders, Organisations, Technical Environments...
- Architecture Views
  - Logical view, Process view, Physical view, Development view + Use case scenarios...
- Architectural activities and process
- Software architecture knowledge areas

Architecture Process



Quality Attributes

- Software Requirements
  - Functional requirements, Quality requirements (NFRs), Constraints
- Quality Attributes
  - Modeling quality attribute scenarios: Source, Stimulus, Artefact, Environment, Response, Measure
  - Availability, Interoperability, Modifiability, Performance, Security, Testability, Usability, X-ability...
  - Tactics for quality attributes
- Architecturally Significant Requirements
  - How to gather and identify ASRs: Requirements, Interviews (QAW), Business goals, Utility tree

Architecture Patterns

- Architecture Patterns
  - Context, Problem, Solution: elements + relations + constraints
- Module Patterns
  - Layered pattern
- Component-Connector Patterns
  - Broker pattern, Model-view-controller pattern, Pipe-and-filter pattern, Client-server pattern, Peer-to-peer pattern, Service-oriented pattern, Publish-subscribe pattern, Share-data pattern
- Allocation Patterns
  - Map-reduce pattern, Multi-tier pattern
- Patterns vs. Tactics

Designing Architecture

- General Design Strategy
  - Abstraction, Decomposition, Divide & conquer, Generation and test, Iteration, Reuse
- Attribute-Driven Design (ADD)
  - Choose a part to design
  - Marshal all ASRs for that part
  - Create and test a design for that part
  - Inputs to and outputs of ADD
  - 8-step process: 1. confirm requirements, 2. choose an element to decompose, 3. identify ASRs, 4. choose a design satisfying ASRs, 5. instantiate elements & allocate responsibilities, 6. define interface, 7. verify & refine requirements, 8. repeat step 2-7 until all ASRs satisfied

Documenting Architecture

Views and Beyond

- Views:
  - Styles (viewpoints), patterns and views
  - Structural views: module views, component-and-connector views, allocation views
  - Quality views
- Documenting views: 1. build stakeholder/view table, 2. combine views, 3. prioritise & stage
- Beyond views: documentation info & architecture info (mapping between views)
- Documentation package: views + beyond

Evaluating Architecture

- ATAM: Architecture Tradeoff Analysis Method
  - Stakeholders involved in ATAM
  - Inputs to and outputs of ATAM
  - Phase 0: Partnership & preparation
  - Phase 1: Evaluation - 1
    - 1. present ATAM, 2. present business drivers, 3. present architecture, 4. identify architectural approaches, 5. generate utility tree, 6. analyse architectural approaches
  - Phase 2: Evaluation - 2
    - 1. present ATAM & results, 7. brainstorm & prioritize, 8. analyse architectural approaches, 9. present results
  - Phase 3: Follow-up

Final Exam

- 简答题、论述题、设计分析题
- 英文题目、中文或英文答题
- 个别题目可能需画图
- 卷面分数 = 基础内容60% + 高阶内容40%
- 总评成绩 = 平时作业40% + 期末考试60%

软件研发效能实验室科研框架