Software Architecture

• Definitions
  – http://www.sei.cmu.edu/architecture/published_definitions.html
  – ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems

  • Architecture is defined by the recommended practice as the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.
Levels of Abstraction

- **Architecture**
  - Highest Level of Abstraction
  - Concerned with quality attributes like performance and reliability, as well as standards and design constraints

- **Design**
  - Few Details

- **Implementation**
  - Complete Details
Typical Elements

- Architecture
- Design
- Implementation

- Deployment Diagrams, Architectural Styles
- Class Diagrams, Design Patterns
- Generics, Inheritance, Annotations, Statements
## Architectural Styles

- Sequential
- Monolithic
- Pipe-and-filter
- Parallel/Distributed
- Layered/n-tier
- State-machine
- Client-Server
- Peer-to-Peer
- Event-Driven
- Component-Based
- Plugin
- Blackboard
- Service-oriented
- Space-based
- Representational State Transfer
- Database-centric
4+1 Architecture Views from the Unified Process

- **Logical View**: Classes, interfaces, collaborations
  - End-user Functionality

- **Implementation View**: Components
  - Programmers
  - Software management

- **Process View**: Active classes
  - System integrators
  - Performance
  - Scalability
  - Throughput

- **Deployment View**: Nodes
  - System engineering
  - System topology
  - Delivery, installation
  - Communication

- **Use Case View**: Use cases
UML Diagrams

• Structural
  – Class, Object, Component, Deployment Diagrams

• Behavioral
  – Use-Case, Activity, Sequence, Communication/Collaboration, Statechart Diagrams
Logical View

- Describes an architecturally significant subset of the design model
- Contains a subset of classes, packages, and use-case realizations
- Concerns the functionality, behavior, use of frameworks and patterns
- Uses Class, Interaction, and State Diagrams
Process View

• Describes threads of control and communication between them
• Concerns the availability, reliability, scalability, performance, synchronization
• Uses Component, Class, and Collaboration Diagrams

[Diagram: TimeTracking Server]
Implementation View

- Describes software component organization
- Concerns team organization and configuration management
- Uses Component Diagrams
Deployment View

- Describes physical network configurations
- Concerns the performance, throughput, fault-tolerance, availability, installation, and maintenance
- Uses Deployment Diagrams
Architecturally Significant Classes
Organized by Package
Deployment View
Each report type will correspond to one JSP.
Architecturally Significant Interactions

1: Create Report
2: 
3: 
4: setWorker
5: setProject
6: setTask
7: getReport
8: 
9: 10: submitQuery
11: 12:
SW Architecture Process

- Chief Architect
  - with Architecture Review Board
- Democratic
  - Design by Committee
- SEI: http://www.sei.cmu.edu/architecture/
  - Architecture Tradeoff Analysis Method (ATAM)
- Visual Architecting Process
  - http://www.bredemeyer.com/howto.htm