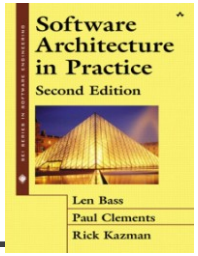
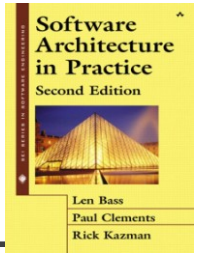


Architecture Design

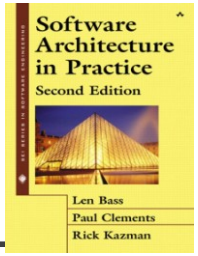
PV167 Project in Object-oriented Design of Information Systems



System Modularization

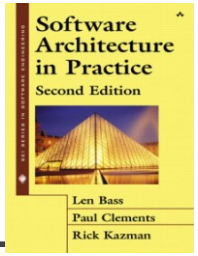
Architecture Design/Part 1

Modularization of Software



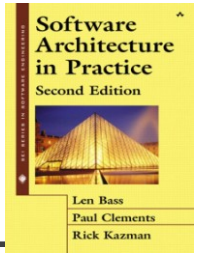
- ✧ Motivation
  - Separation of concerns
  - Break-down of system complexity
  - Reuse
- ✧ Architecture design
  - UML Package diagram (Analysis)
  - UML Component diagram (Design/Implementation)
  - UML Deployment diagram (Realisation)

Implementation-oriented Modularization

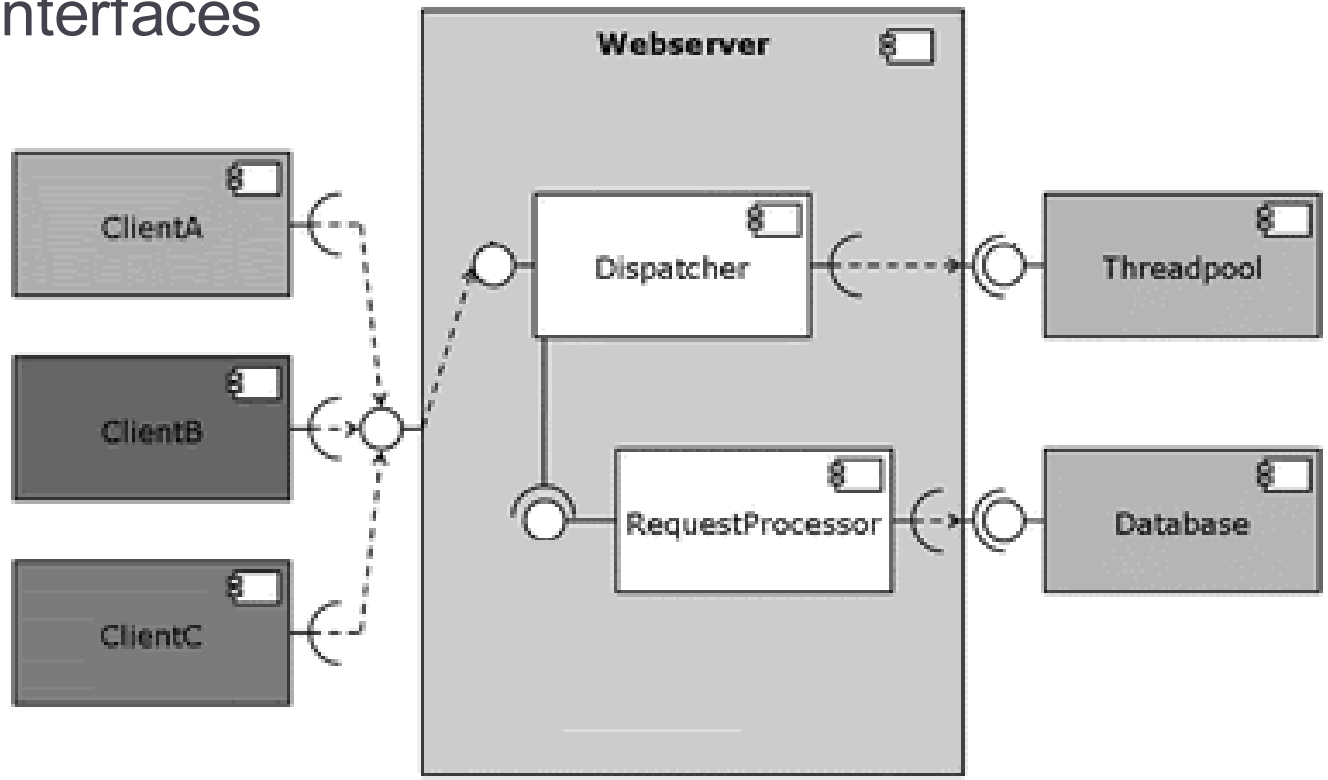


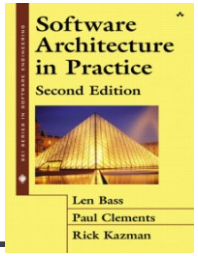
- ✧ Motivation
  - Reuse
  - Architectural tactics
  - Separation of concerns
- ✧ Implementation
  - Service-oriented approach
  - Data passing, i.e. we can no longer rely on shared knowledge
- ✧ Challenges
  - Shared knowledge that is hard to pass along
  - Crosscutting concerns

Components and Interfaces



- ✧ Provided interfaces
- ✧ Required interfaces

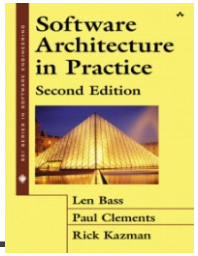




Architectural Patterns and Tactics

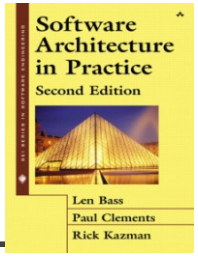
Architecture Design/Part 2

Architectural Patterns



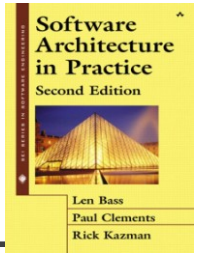
- The Model-View-Controller (MVC) pattern
  - Separates presentation and interaction from the system data.
- The Layered architecture pattern
  - System organised into layers with interfaces to other layers.
- The Repository architecture pattern
  - Interaction through a central repository.
- The Client-server architecture pattern
  - Distribution of data and processing across stand-alone service-providing servers and clients calling the services.
- The Pipe and filter architecture pattern
  - Functional transformations process their inputs to outputs.

Architectural Tactics for Dependability



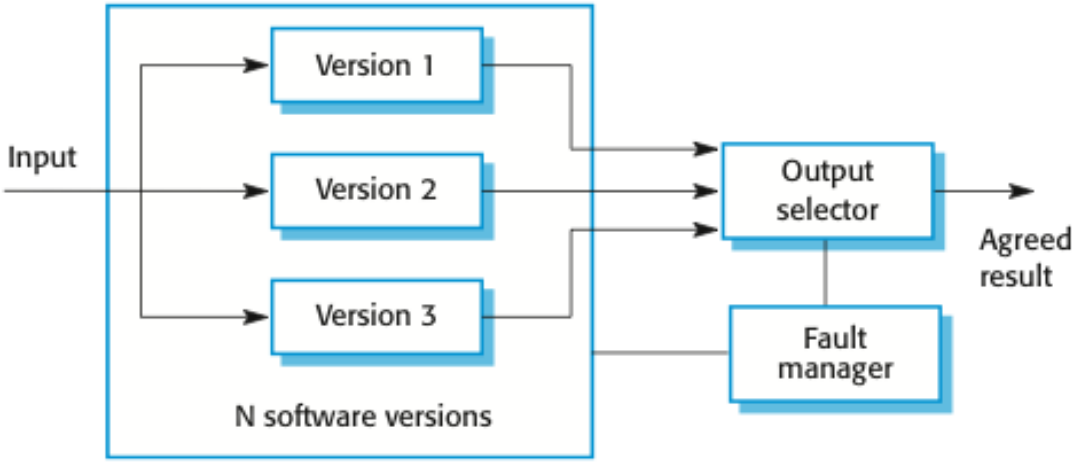
- Redundancy
  - Keep more than 1 version of a critical component available so that if one fails then a backup is available.
  - E.g. switch to backup servers automatically if failure occurs.
- Diversity
  - Provide the same functionality in different ways so that they will not fail in the same way.
  - E.g. different servers may be implemented using different operating systems (e.g. Windows and Linux).
- However, adding diversity and redundancy adds complexity and this can increase the chances of error.

N-version programming pattern

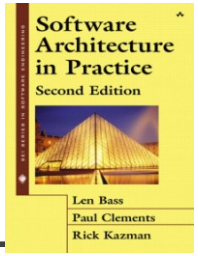


- Multiple versions of a software system carry out computations at the same time.
- The results are compared using a voting system and the majority result is taken to be the correct result.

Which of the tactics are involved here?

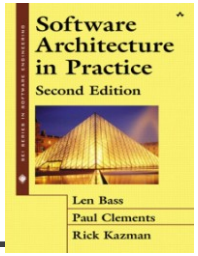


Architectural Tactics for Security

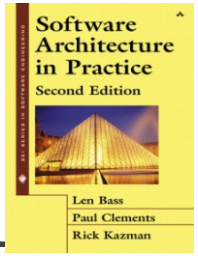


- Use redundancy and diversity to reduce risk
  - Keep multiple copies of data and use diverse infrastructure so that an infrastructure vulnerability cannot be the **single point of failure**.
- Compartmentalize your assets
  - Organize the system so that assets are in separate areas and users only have access to the information that they need rather than all system information.
- Design for recoverability
  - Design the system to simplify recoverability after a successful attack.

Architectural Tactics for Performance



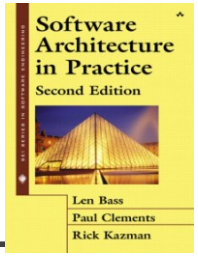
- Introduce concurrency
  - If requests can be processed in parallel, the blocked time can be reduced.
- Maintain multiple copies of either data or computations
  - The purpose of replicas is to reduce the contention that would occur if all computations took place on a central server.
- Increase available resources
  - Faster processors, additional processors, additional memory, and faster networks all have the potential for reducing latency.



Architectural Quality and its Prediction

Architecture Design/Part 3

Quality prediction



- Palladio Component Model
  - Quality prediction and design simulation framework