Software Design and Architecture

Lecture 1

Prepared by

Soha Makady

Outline

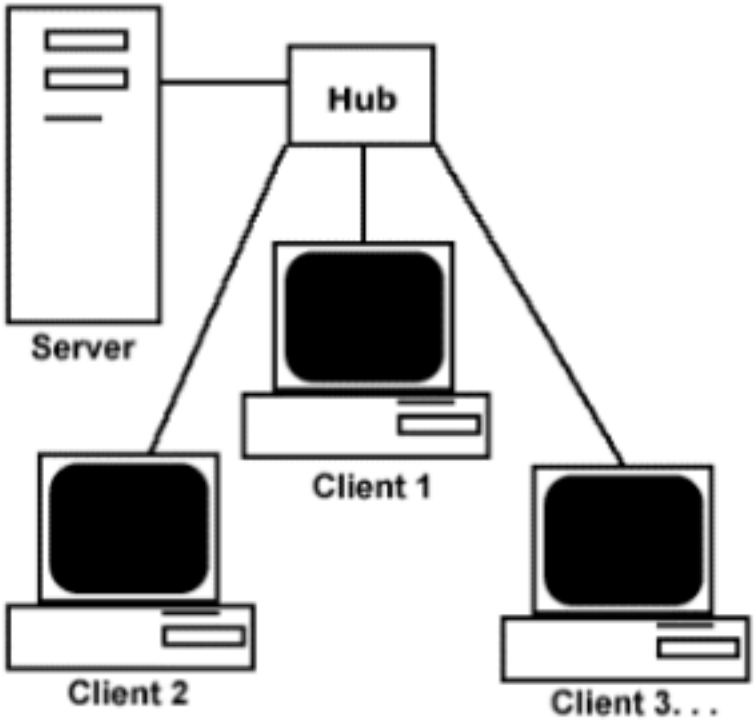
- A tale of two systems
- Rackspace real case study
- Software Architecture versus Software Design
- Course Organization

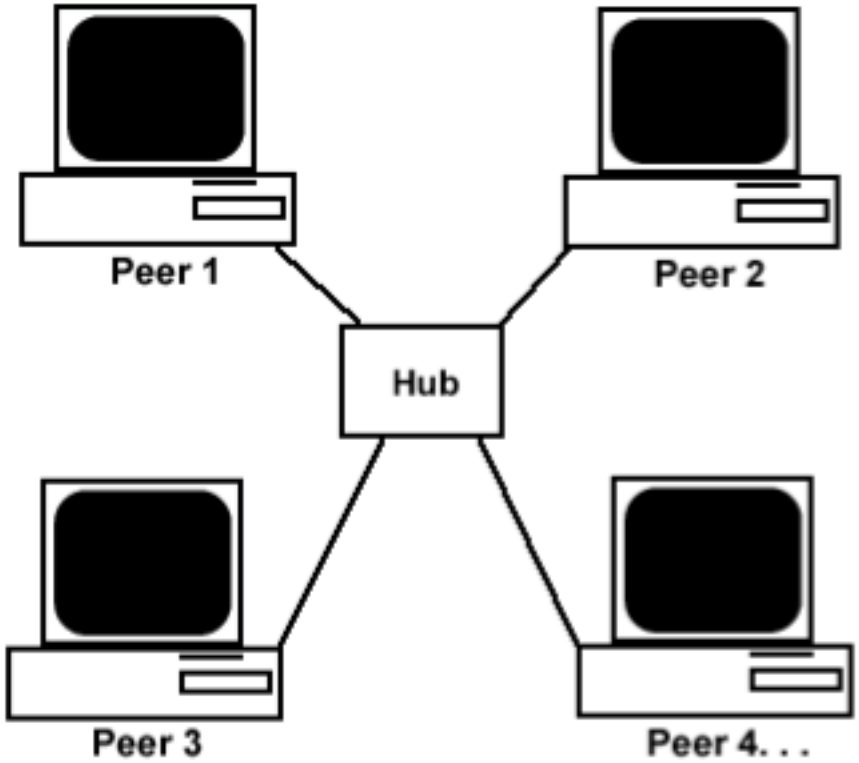
Outline

- **<u>A tale of two systems</u>**
- Rackspace real case study
- Software Architecture versus Software Design
- Course Organization

A Tale of Two Systems

- Have you ever used any VoIP system (e.g., Skype)?
- Why do you use such systems, rather than regular phones?





Based on George FairBanks talk on Software Architecture

A Tale of Two Systems

Plain Old Telephone System

- Feature:
  Call subscriber
- Architecture:
  Centralized hardware switch
- Good qualities
  Works during power outages
  Reliable
  Emergency calls get location information

Same feature

Skype

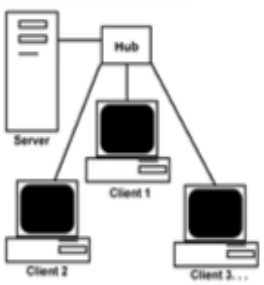
- Feature:
  Call subscriber
- Architecture:
  Peer-to-peer software
- Good qualities
  Scales without central hardware changes
  Easy to add new features (e.g., video calling)

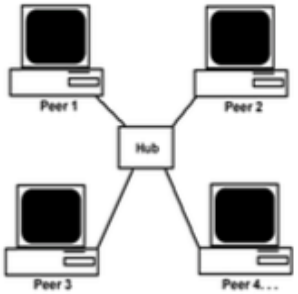
Architects pay more attention to **qualities** that arise from architecture choices.

Based on George FairBanks talk on Software Architecture

Two Telephone Systems – Pro's and Con's

Which one is better?





Quality Attribute	Landline Phone	VoIP (Skype)
Power Outage Tolerant	++	--
Reliable	+	-
Scalable	-	++
Extendable (new features)	-	++

Based on Dietmar Pfhal Lecture on Software Architecture

Trade-Offs and Decision-Making

Telephone system for a Fire Brigade Station:

Which one is better?

Quality Attribute	Landline Phone	VoIP (Skype)
Power Outage Tolerant	++	--
Reliable	+	-
Scalable	-	++
Extendable (new features)	-	++

Trade-Offs and Decision-Making – Template for capturing design rationales:

Telephone system for a Fire Brigade Station:

- Because <Quality Attribute 1> is more important than <Quality Attribute 2> for this system, we choose <technical (design/architecture) option>, accepting <drawback>

Quality Attribute	Landline Phone	VoIP (Skype)
Power Outage Tolerant	++	--
Reliable	+	-
Scalable	-	++
Extendable (new features)	-	++

Trade-Offs and Decision-Making – Template

Telephone system for a Fire Brigade Station:

- Because <Power Outage Tolerance> is more important than <Scalability> for this system, we choose a <Landline Phone>, accepting <a higher cost for adding new subscribers>.

Quality Attribute	Landline Phone	VoIP (Skype)
Power Outage Tolerant	++	--
Reliable	+	-
Scalable	-	++
Extendable (new features)	-	++

Outline

- A tale of two systems
- **<u>Rackspace real case study</u>**
- Software Architecture versus Software Design
- Course Organization

Rackspace *Real* Case Study

- Rackspace is a real company that manages hosted email servers.

CSR:
Customer Service Request

CSR desktop

Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study

- Here's the situation
  - You are a hosting provider
  - You rent mail servers
  - Customers have problems
  - You use the mail log files to diagnose their problems
- The big question:
  - How would you build it?
- Let's assume you can build it
  - … but different architectures yield different qualities

Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study

- Why is this hard?
  - You have hundreds of servers
  - You generate GBs of logs daily
  - Collecting logs takes time
  - Searching logs takes time
- Hints and options
  - Central collection of logs?
  - Distributed searching of logs?
  - Can you pre-process logs to speed up queries?

Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study - Architecture 1

- Hosting provider of email service
- Email log files

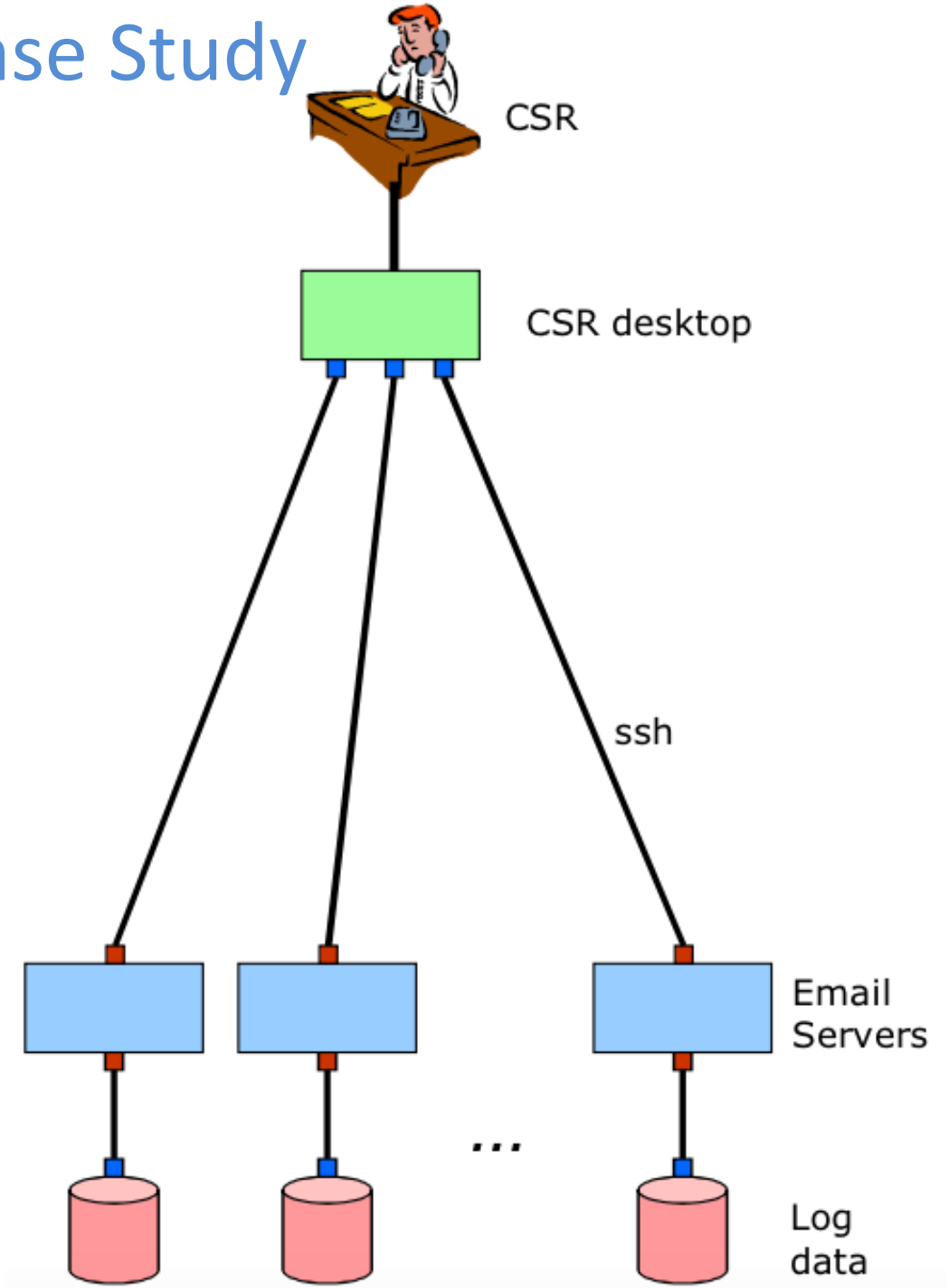
- Task: debug user problem

- Architecture
  - CSR desktop computer
  - ssh connections to servers
  - Servers with local log files

- Procedure
  - Write query as grep expression
  - Script runs via ssh on every server
  - Results aggregated

Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study Architecture 1



Based on George FairBanks talk on software Architecture

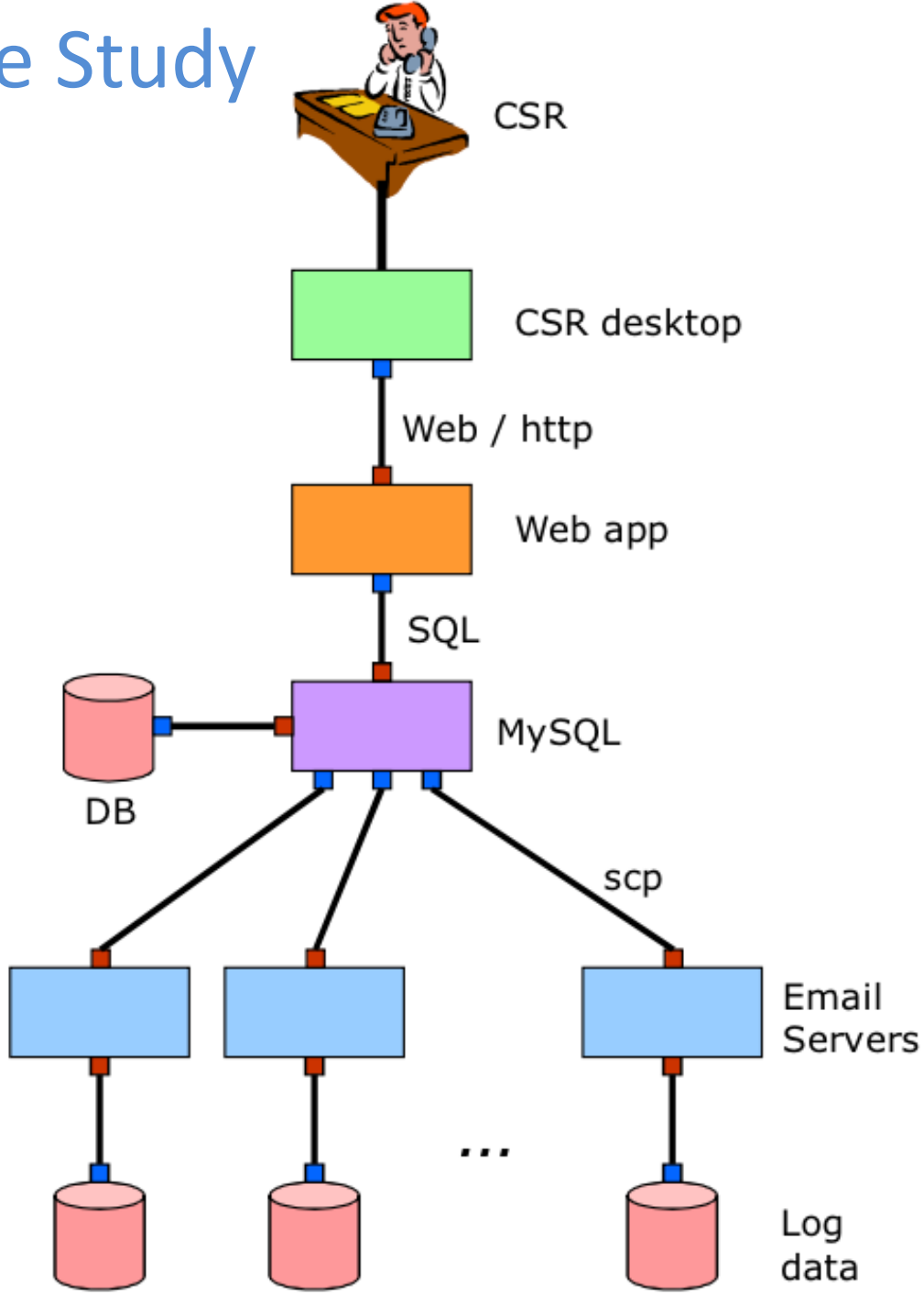
Rackspace *Real* Case Study - Architecture 2

- Architecture
  - CSR desktop computer
  - Web application
  - MySQL database
  - scp log transfer
  - Servers with local log files

- Procedure
  - Every 10 minutes, send log files to MySQL server; delete original
  - Parse and load logs into MySQL
  - Combine new logs with old
  - Send query to MySQL server; answered from DB data

Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study - Architecture 2



Based on George FairBanks talk on software Architecture

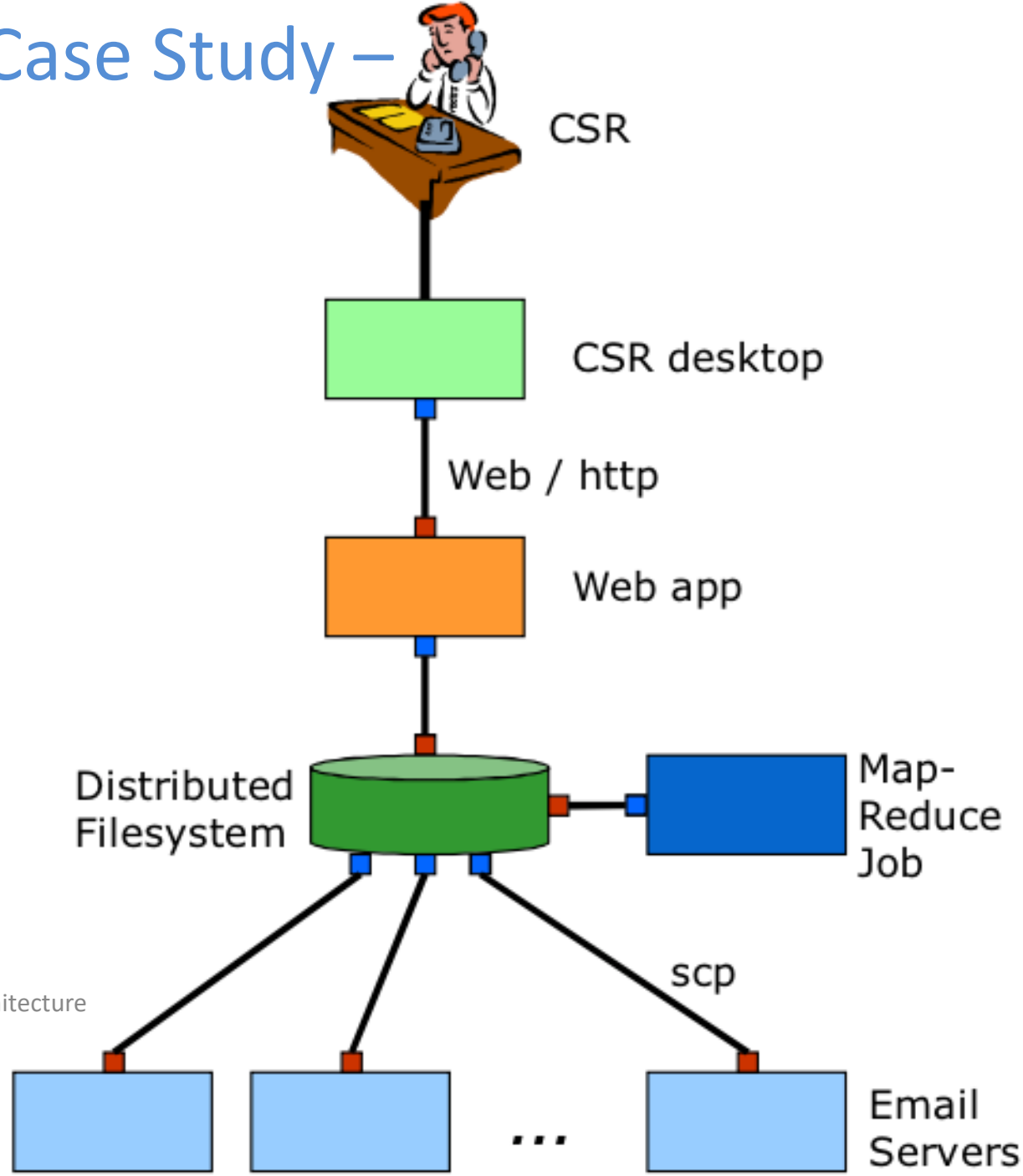
Rackspace *Real* Case Study - Architecture 3

- Architecture
  - CSR desktop computer
  - Web application
  - Distributed filesystem
  - Map-Reduce job cluster
  - Servers with local log files

- Procedure
  - Log data continuously streamed from email servers to distributed filesystem (HDFS)
  - Every 10 minutes, Map-Reduce job runs to process log files, create index
  - Web app queries index

Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study – Architecture 3



Based on George FairBanks talk on software Architecture

Rackspace *Real* Case Study – Tradeoffs

- Tradeoff: Data freshness
  - V1: Queries run on current data
  - V2: Queries run on 10 minute old data
  - V3: Queries run on 10-20 minute old data

- Tradeoff: Scalability
  - V1: Noticeable email server slowdown (dozens of servers)
  - V2: MySQL speed/stability problems (hundreds of servers)
  - V3: No problems yet

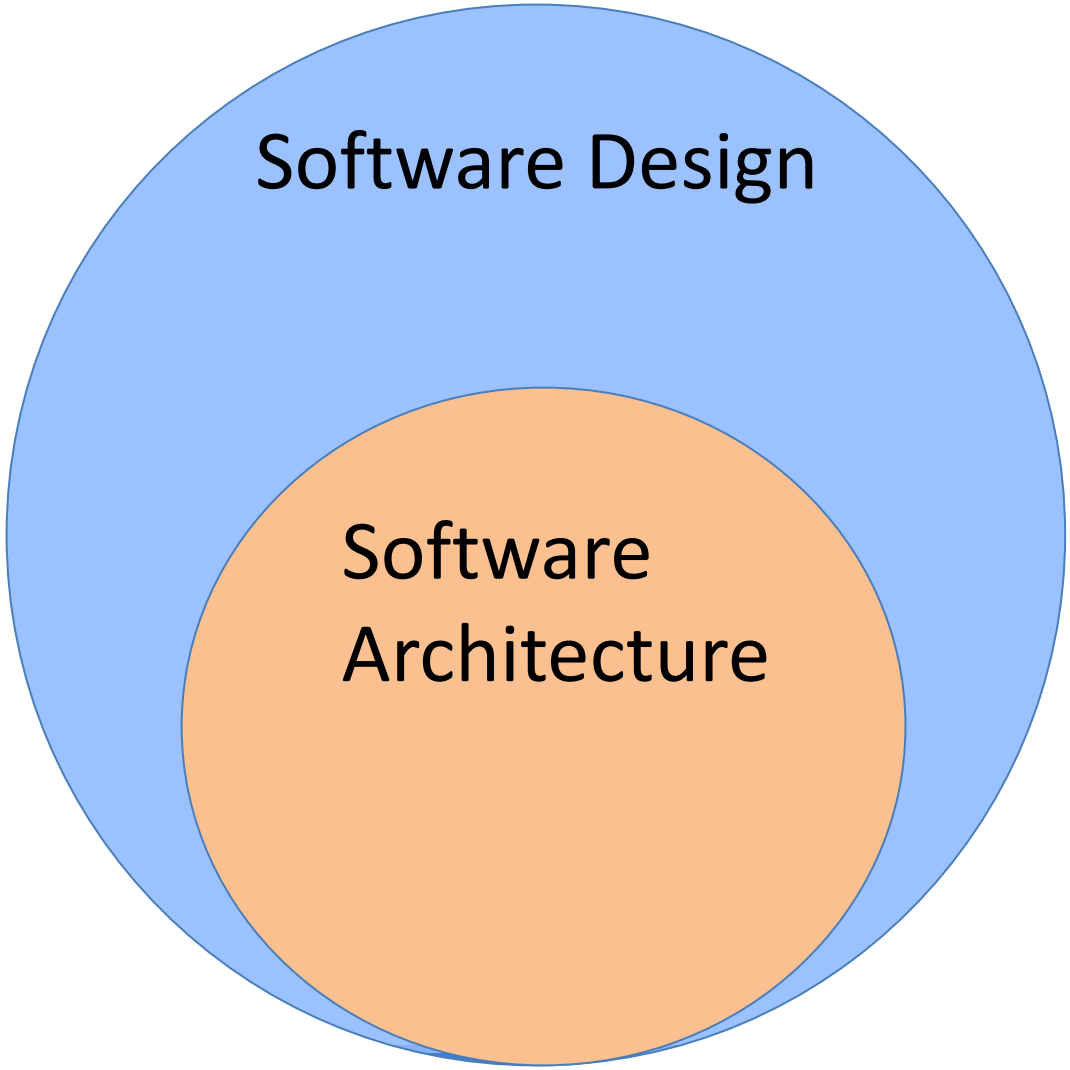
Outline

- A tale of two systems
- Rackspace real case study
- **Software Architecture versus Software Design**
- Course Organization

Software Architecture vs. Software Design

- What is Software Architecture and how does it relate to Software Design?
- Let us see a definition …
- The software architecture of the system is the **set of structures** needed to **reason about the system**, which comprises software elements, relations among them, and properties of both

Software Architecture vs. Software Design



Software Architecture vs. Software Design

Possible failures:

Crashes

Insecurity

Long response time

Not portable

Not extensible

Software developer

Based on George FairBanks talk on Software Architecture

Software Architecture vs. Software Design

Feared failures:

Crashes

Insecurity

Long response time

Not portable

Not extensible

Possible designs:

Swing

Amazon Cloud

Relational DB

Hadoop

Software developer

Based on George FairBanks talk on Software Architecture

Software Architecture vs. Software Design

Feared failures:

Crashes

Insecurity

Long response time

Not portable

Not extensible

Selected designs:

Swing

Amazon Cloud

Relational DB

Hadoop

Software developer

Based on George FairBanks talk on Software Architecture

Software Architecture vs. Software Design

***Design** details*

Complex System

***Architecture** details*

Architecture Model

Based on George FairBanks talk on Software Architecture

Software Architecture vs. Software Design

- The details that you include in your **architecture** model to reason about the failures are architectural details.
- The full design of the system includes other details necessary to build the system, ***but not needed to reason about the failures.***
  - *Such details are* ***design details****.*

Software Architecture vs. Software Design

- Let us look again at the same definition…
- The software architecture of the system is the **set of structures** needed to **reason about the system**, which comprises software elements, relations among them, and properties of both

Software Architecture vs. Software Design

- Architecture is design, but not all design is architecture.
- Architecture consists of architectural design decisions, and all others are non-architectural.
  - What decisions are non-architectural?
  - What design decisions does the architect leave to the discretion of others?
  - How about detailed decisions that are still architectural?

Stack or Queue?