Informatics 43

LECTURE 7

“HOW DO WE STRUCTURE THE SOFTWARE?”

Last lecture

- Use case = textual description defining interactions between an actor and the system to achieve the primary actor’s goal
  - Includes different flows (basic, alternative, exception)
- Use case model
  - Diagrams (actors, use cases, relationships, system boundary)
  - Use case descriptions
- Use cases serve as a unifying thread throughout development
- Use cases serve as a communication/understanding tool among diverse stakeholders

Today’s lecture – **How do we structure the software?**

- Introduction
- Defining software architecture
- Architecture in action
- Software architecture’s elements
- Architectural erosion

Some slides adopted and adopted from “Software Architecture: Foundations, Theory, & Practice” by Taylor, Medvidovic, and Dashofy

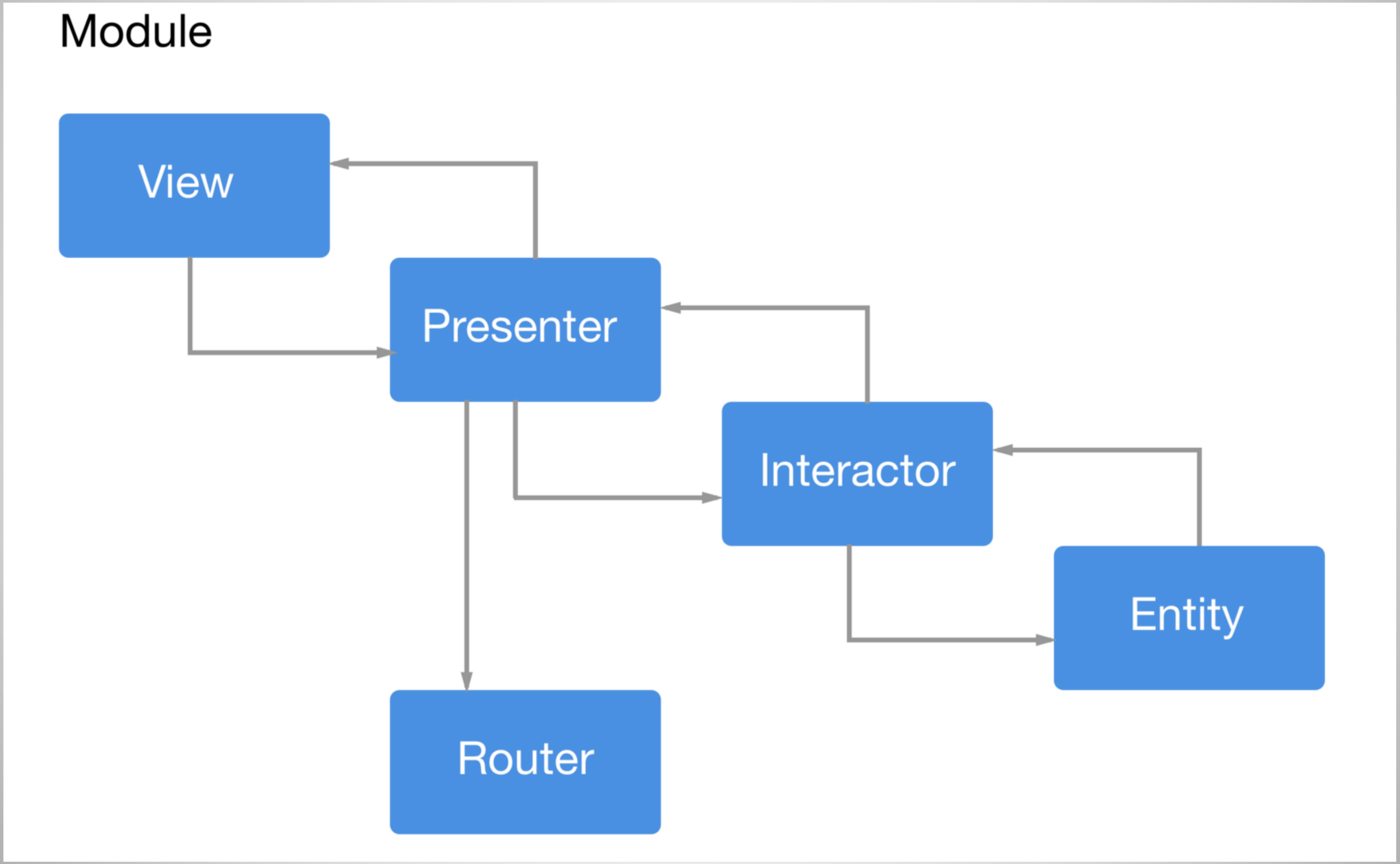
Today's lecture – **How do we structure the software?**

- Introduction
- Defining software architecture
- Architecture in action
- Software architecture's elements
- Architectural erosion

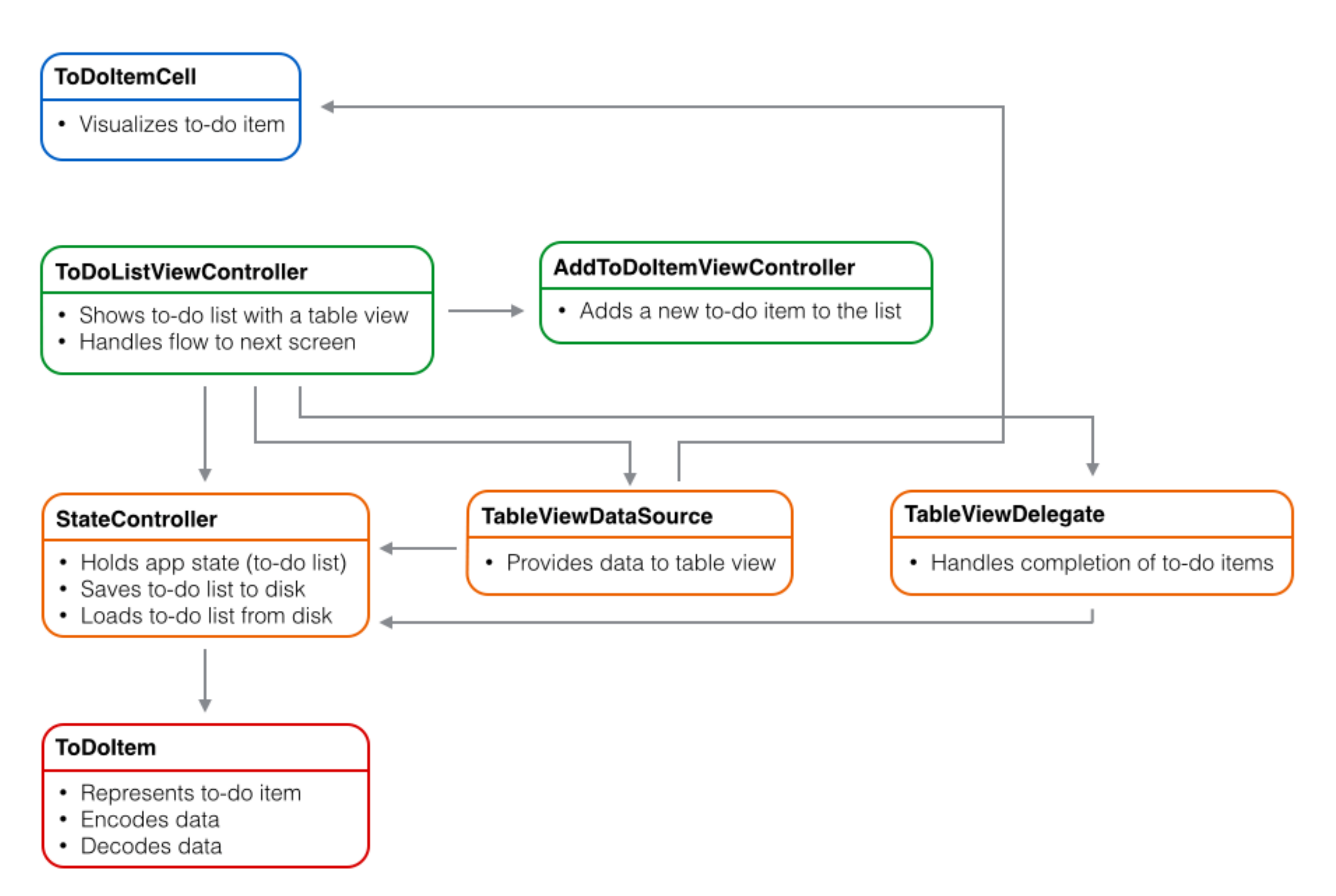
Revisiting our ToDoList App

- We’ve got enough requirements; now what?

One Possibility



Another Possibility



ToDoItemCell

- Visualizes to-do item

ToDoListViewController

- Shows to-do list with a table view
- Handles flow to next screen

AddToDoItemViewController

- Adds a new to-do item to the list

StateController

- Holds app state (to-do list)
- Saves to-do list to disk
- Loads to-do list from disk

TableViewDataSource

- Provides data to table view

TableViewDelegate

- Handles completion of to-do items

ToDoItem

- Represents to-do item
- Encodes data
- Decodes data

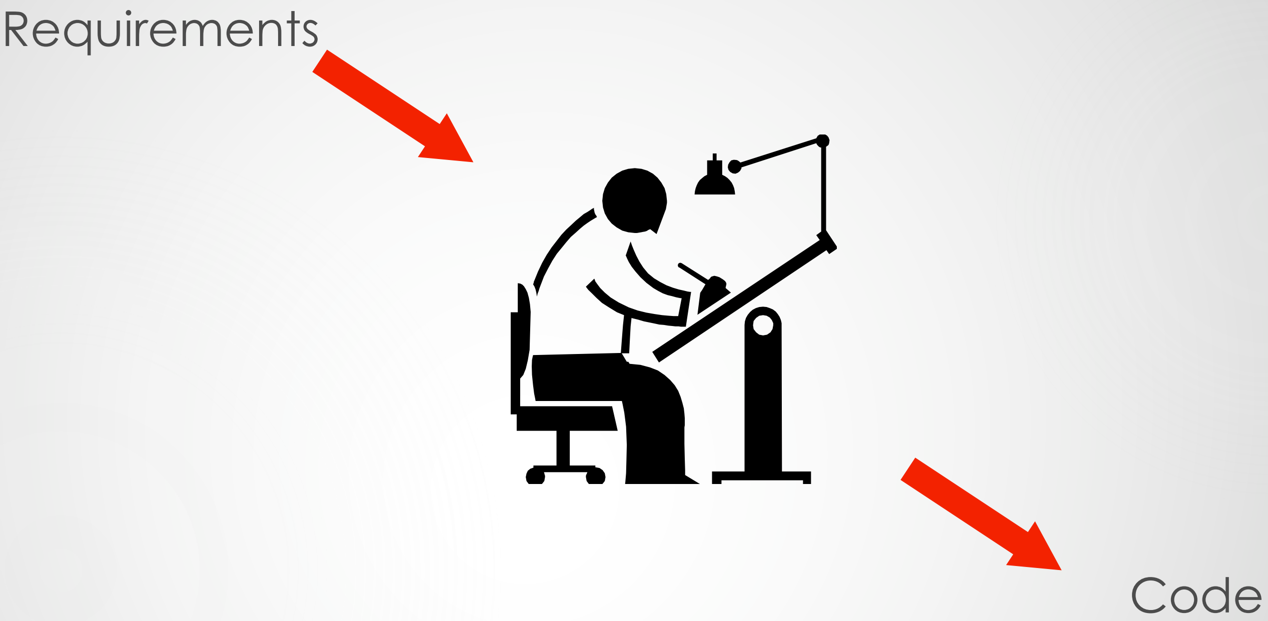
Source: https://www.smashingmagazine.com/2016/05/better-architecture-for-ios-apps-model-view-controller-pattern/

Software Architecture

Requirements

Code

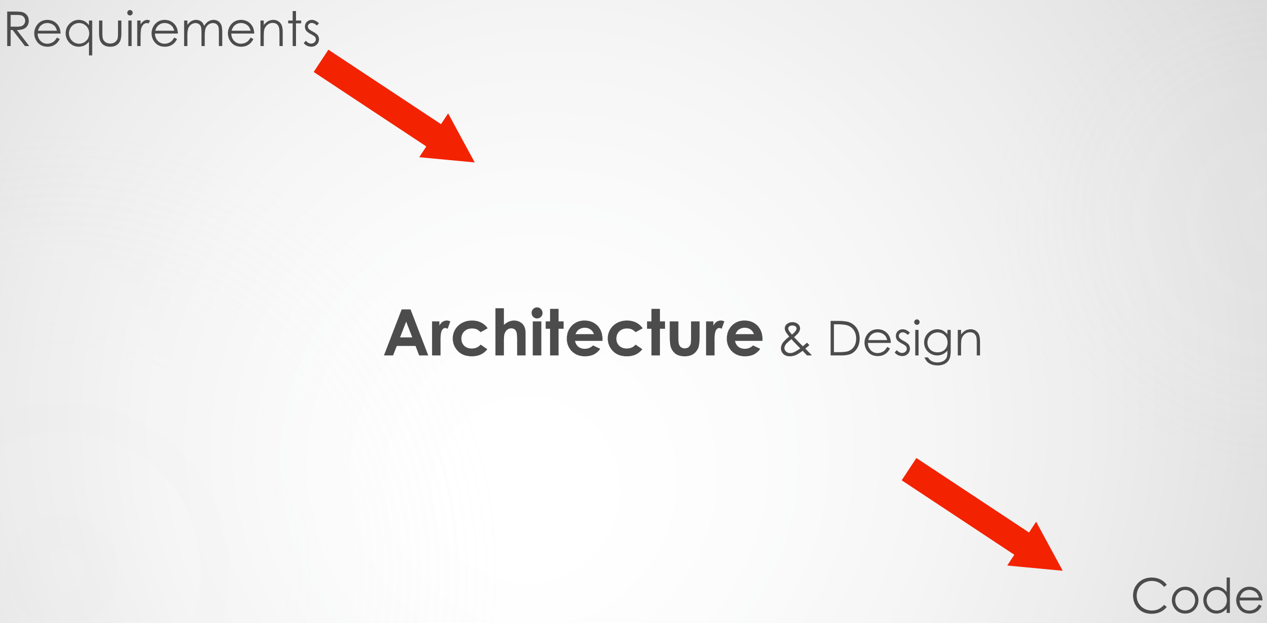
Software Architecture



Requirements

Code

Software architecture



Requirements

Architecture & Design

Code

An analogy to building architectures

The architect

- A distinctive role and character in a project
- Broad training
- Extensive experience
- Deep understanding of the domain
- Leads the team
- Good communicator
- Decision maker
- Deals with a higher level of abstraction than those performing construction
- Often serves as the interface to key business stakeholders/customers

Limitations of analogy

- The nature of software is different from that of building architecture
  - Software serves a much broader range of purposes
  - We know much more about buildings than software
  - Software is much more malleable than physical materials
  - Software is a machine
  - Software is invisible

Why architecture in software engineering?

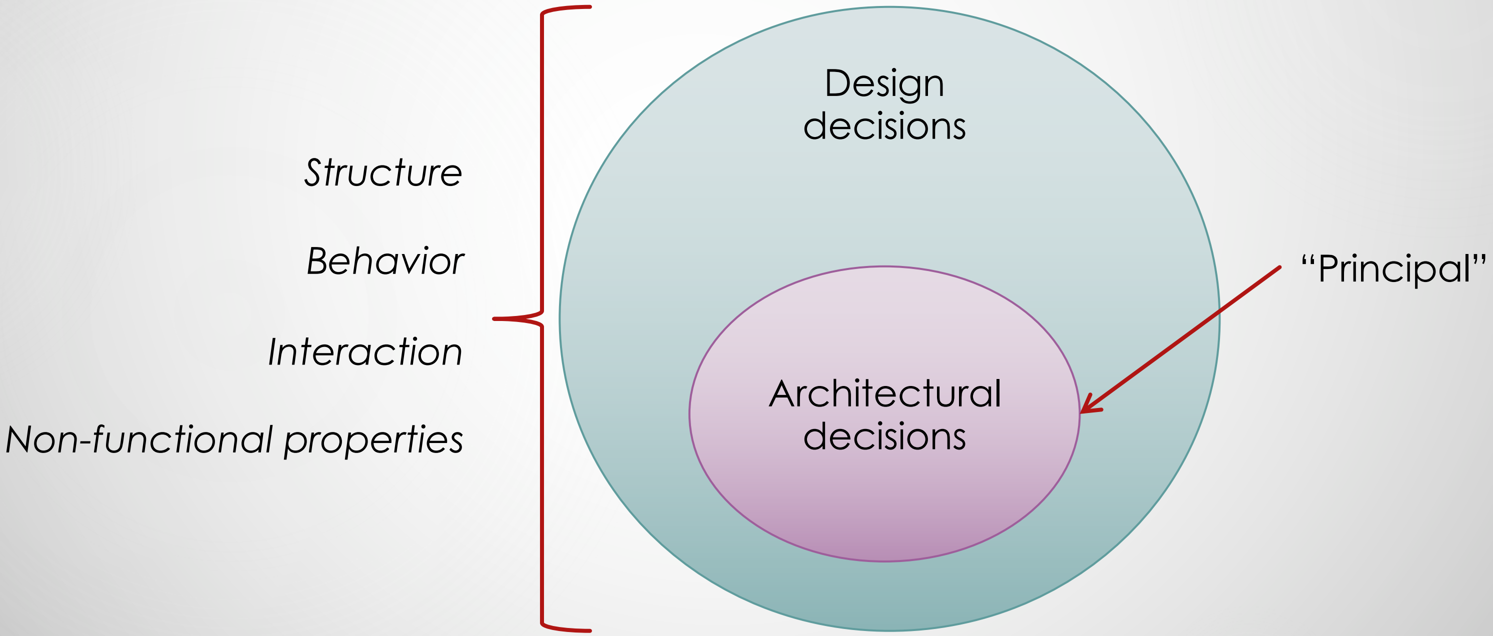
- Intellectual control
- Conceptual integrity
- Effective project communication
- Reusability
- Maintainability (management of a set of variant systems)

Today’s lecture – **How do we structure the software?**

- Introduction
- Defining software architecture
- Architecture in action
- Software architecture’s elements
- Architectural erosion

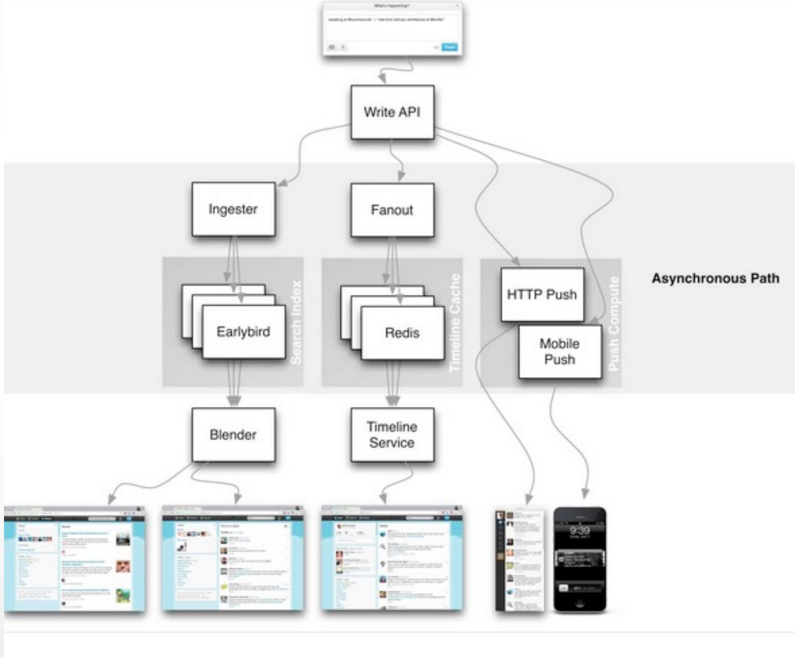
Defining software architecture

A software system's architecture is the set of *principal design decisions* about the system



Defining software architecture

Software architecture is the blueprint for a software system’s construction and evolution



Other definitions

“The set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both.” [Clements, Bass, Kazman]

Other definitions

"How your whole system (in the widest possible sense) will be decomposed into processes or services; how data are stored, communicated, and processed; and how all parts fit together to deliver the required functionality, reliability, capacity, scalability, maintainability, and portability." [Spinellis]

Other definitions

*"The **highest-level** breakdown of a system into its **parts**; the **decisions** that are hard to change." [Fowler]*

Other definitions

"The clear definition of multiple high-level components that, when working together, form your system and solve your problem." [Natarén]

Attendance Quiz

Definition of Software Architecture

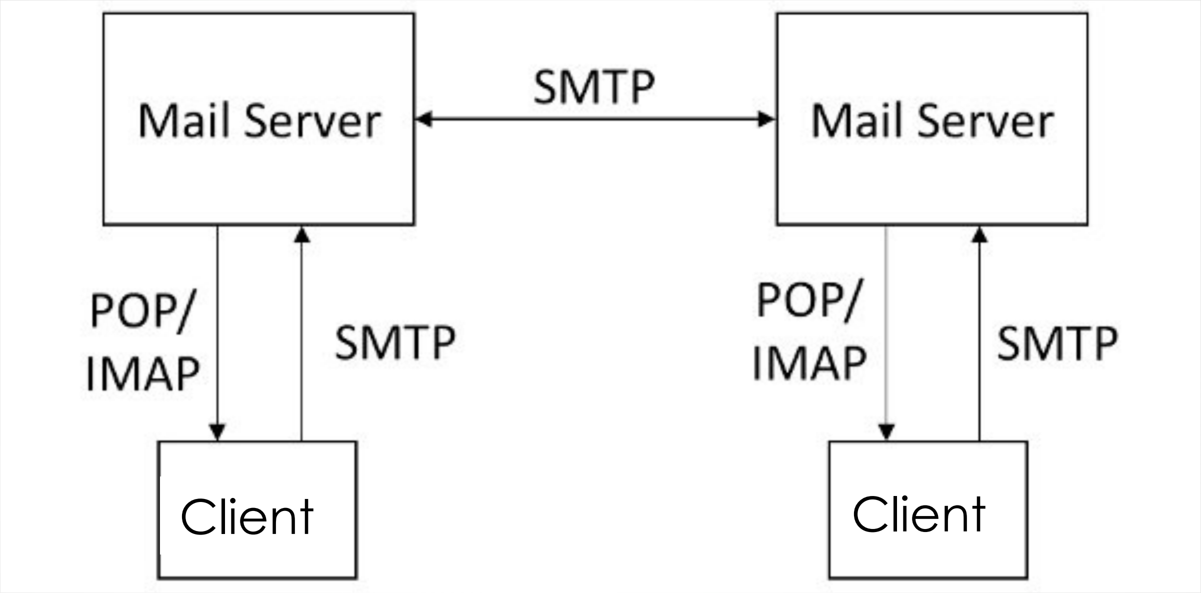
Today's lecture – **How do we structure the software?**

- Introduction
- Defining software architecture
- Architecture in action
- Software architecture's elements
- Architectural erosion

Email: functional specification

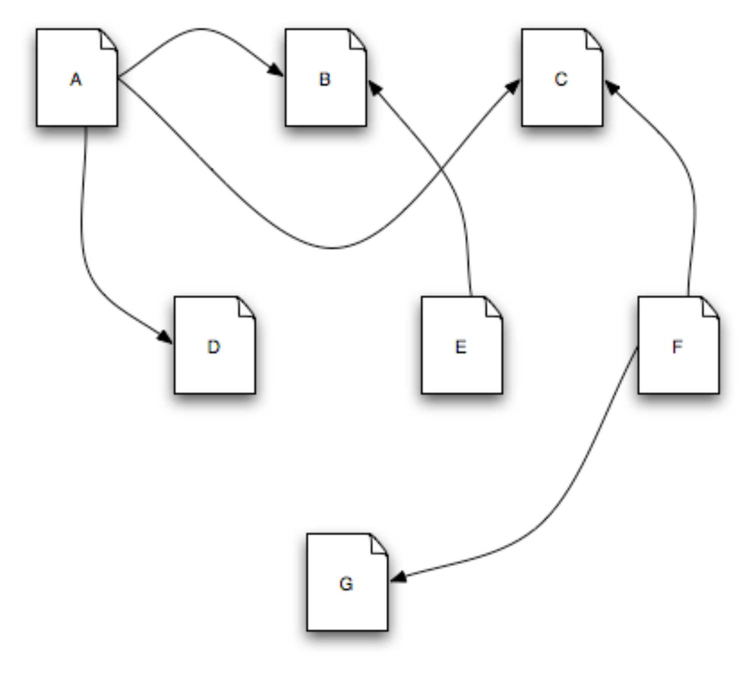
- Send and receive messages
- Get your own messages, not others'
  - Addressing scheme
- Store messages
- Electronic
- Blocks spam

Email - Architecture



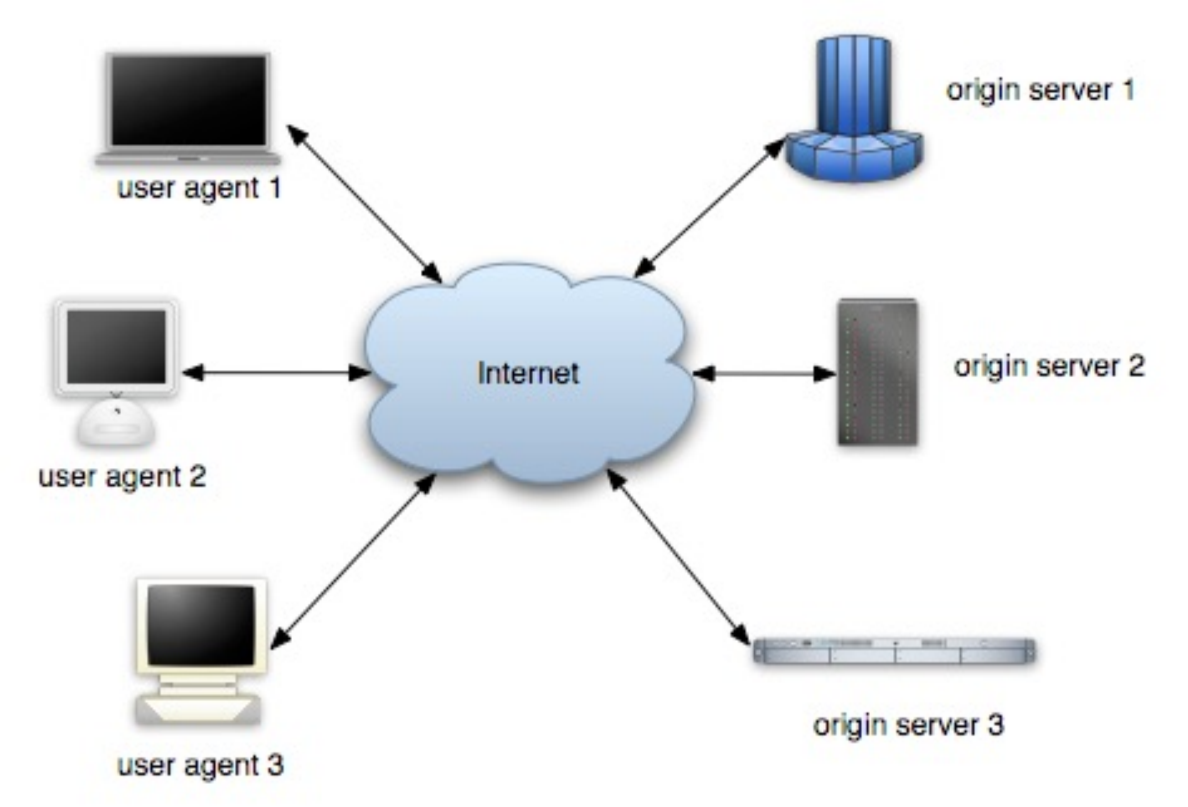
Architecture in action: WWW

This is one way to describe the Web



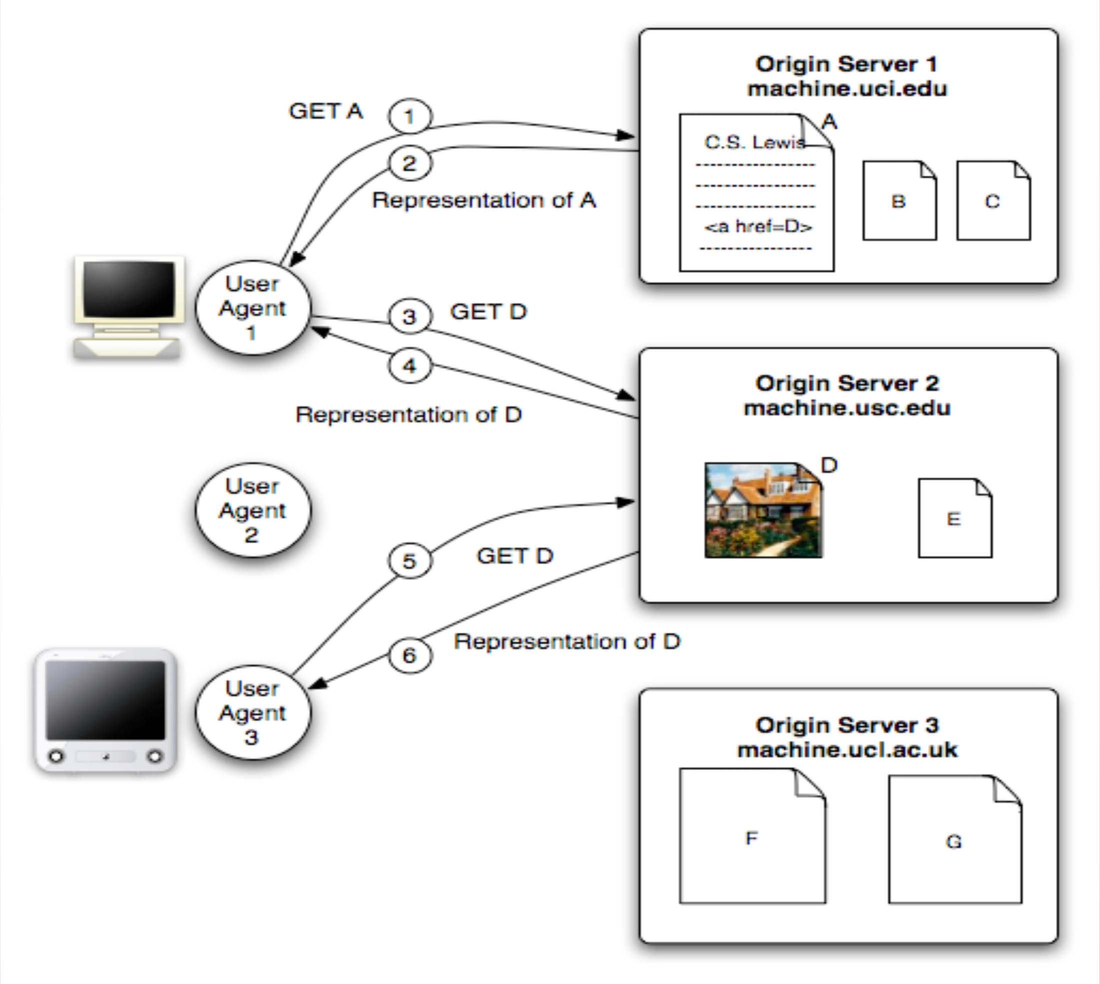
Architecture in action: WWW

So is this



Architecture in action: WWW

And this



WWW in a (Big) Nutshell

- The Web is a collection of resources, each of which has a unique name (URL)
- A URL can be used to determine the identity of a machine on the Internet (origin server), from which the resource may be obtained
- Clients (user agents/Web browsers) make requests of servers for their resources

WWW in a (big) nutshell (continued)

- Clients manipulate representations of resources
- All communication between user agents and origin servers must be performed by a simple, generic protocol (HTTP)
- All communication between user agents and origin servers must be fully self-contained

Observations about WWW’s architecture

- There is no single piece of code that implements the architecture
- Stylistic constraints of the Web’s architectural style are not apparent in the code
- **One of the world’s most successful applications is only understood adequately from an architectural vantage point**

Observations about WWW’s architecture

- There is no single piece of code that implements the architecture
- Stylistic constraints of the Web’s architectural style are not apparent in the code
- **One of the world’s most successful applications is only understood adequately from an architectural vantage point**

All of the diagrams and text on the previous several slides
= WWW’s architecture
(also known as the REST reference architecture)

Facebook Architecture

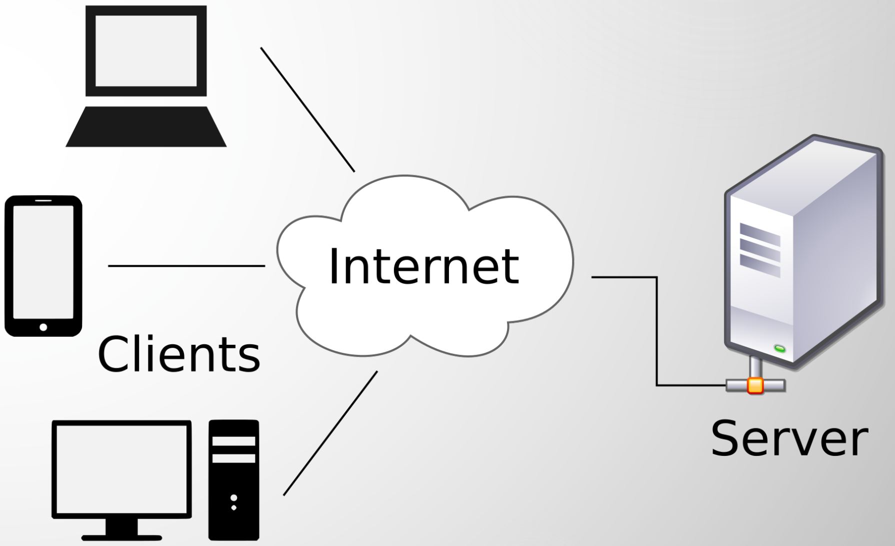
- Functional requirements?
  - Friend lists
  - Status updates
  - News feed
  - Comments/likes/reactions
  - Messaging
  - Photos
  - Location services
  - 3rd party apps
  - …

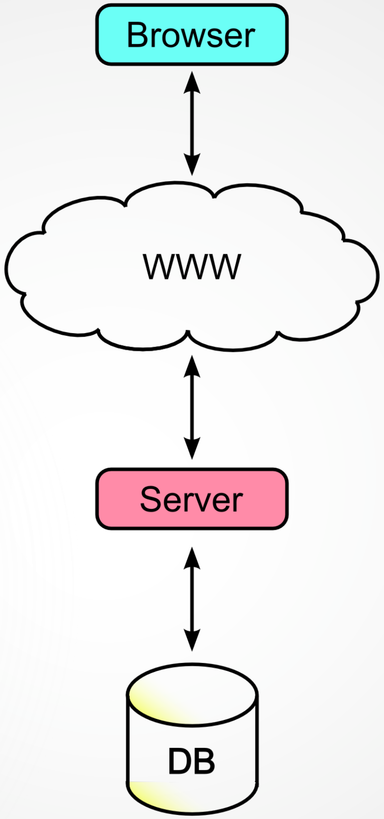
Facebook Architecture

- Non-functional requirements?
  - Efficiency/performance
  - Scalability
  - Availability
  - Security
  - Privacy
  - Reliability
  - Portability

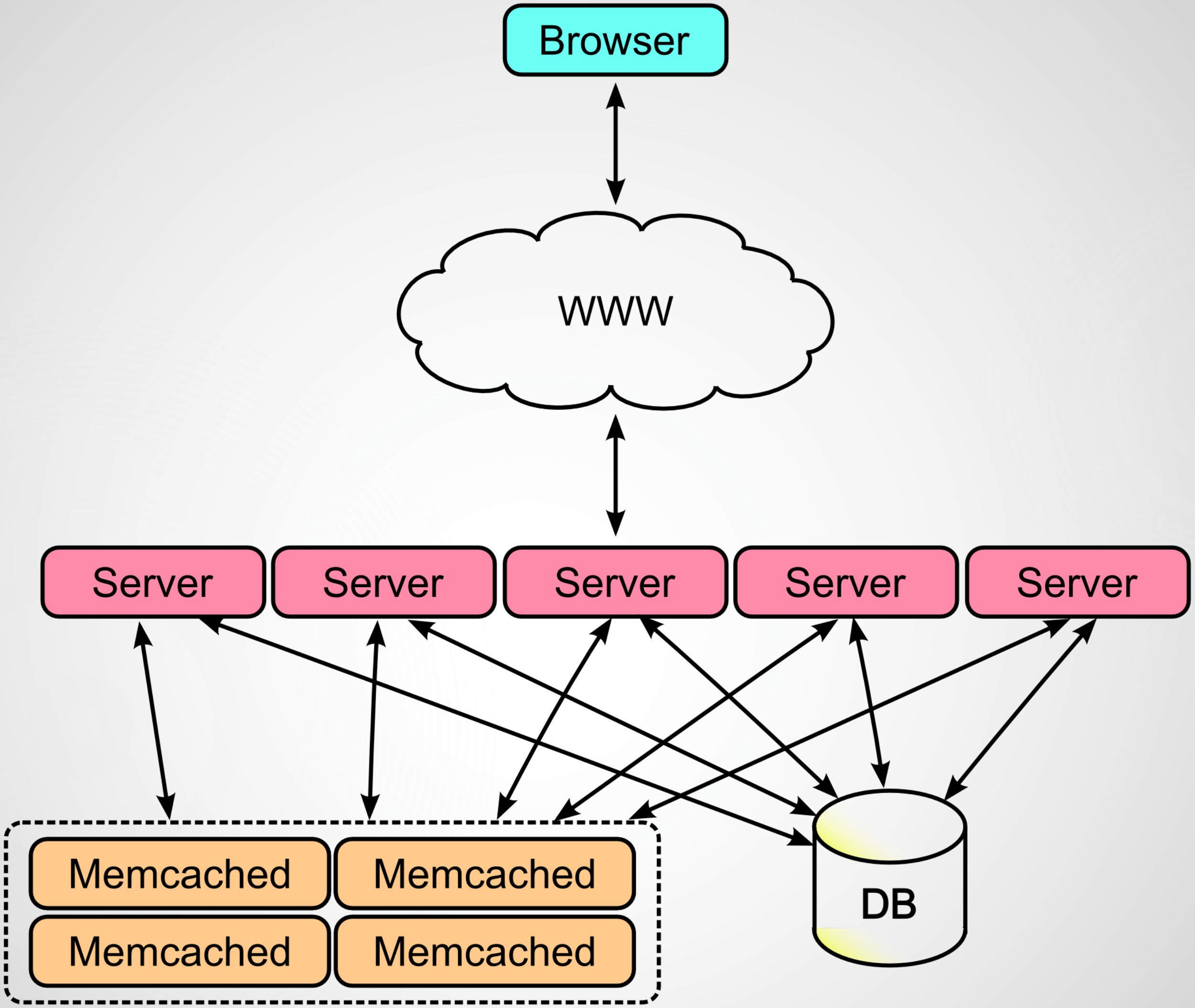
Client Server Architecture

- Most common distributed system architecture
- Clients make requests of servers
- Centralized
  - data
  - security
  - access

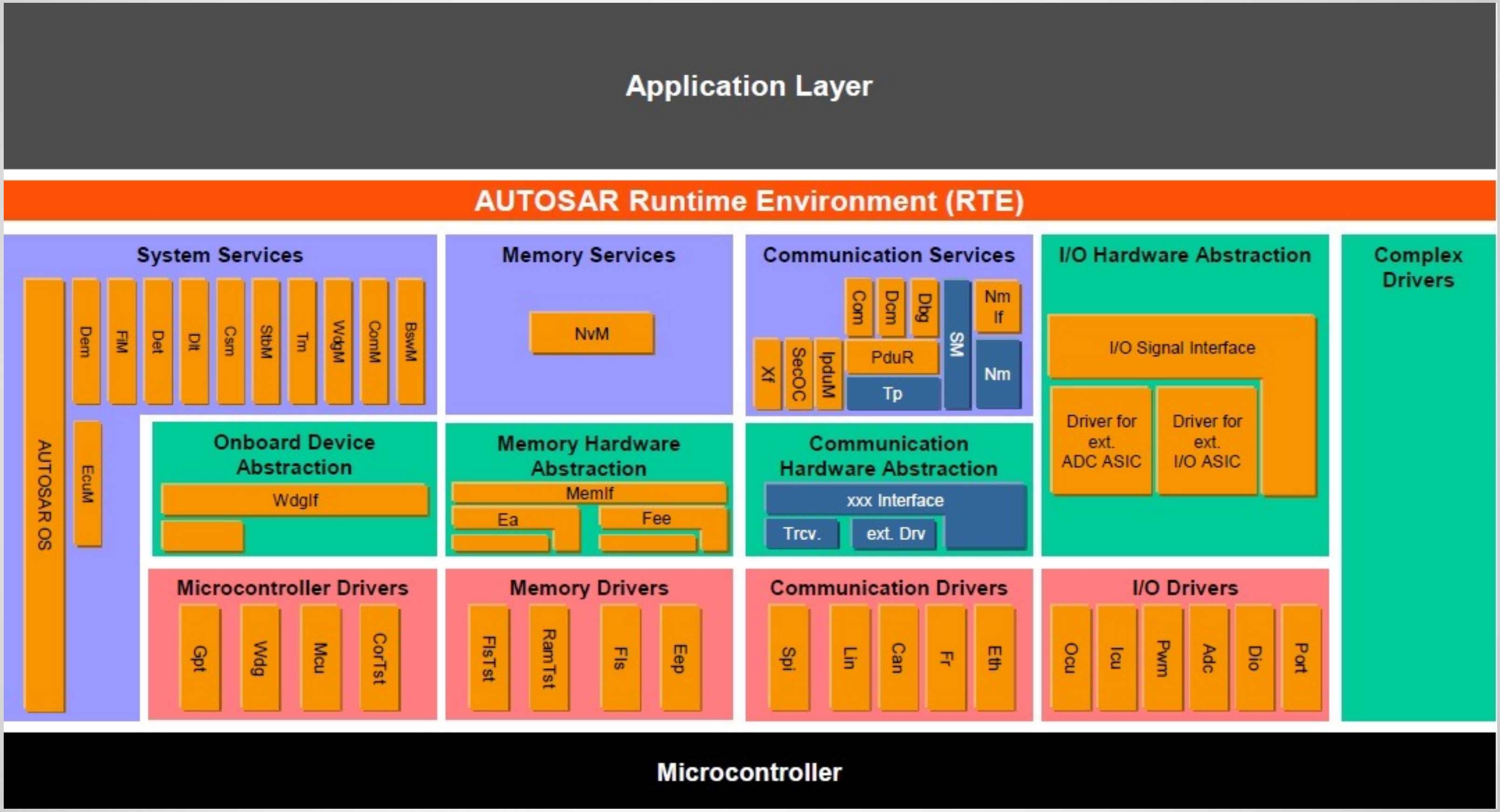




Browser
WWW
Server
DB

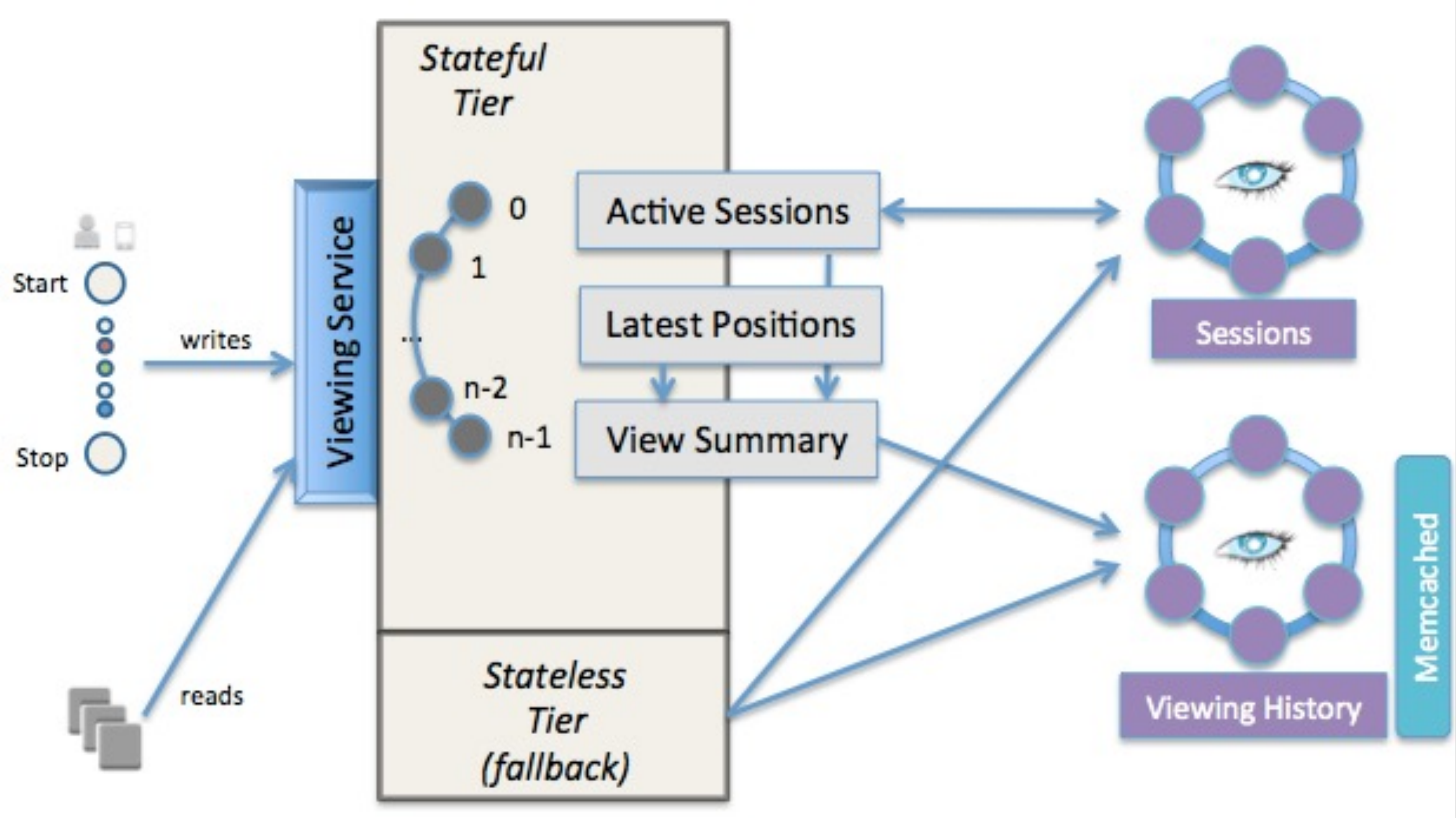

Browser
WWW
Server
Server
Server
Server
Server
Memcached
Memcached
Memcached
Memcached
DB

AUTOSAR Architecture Diagram

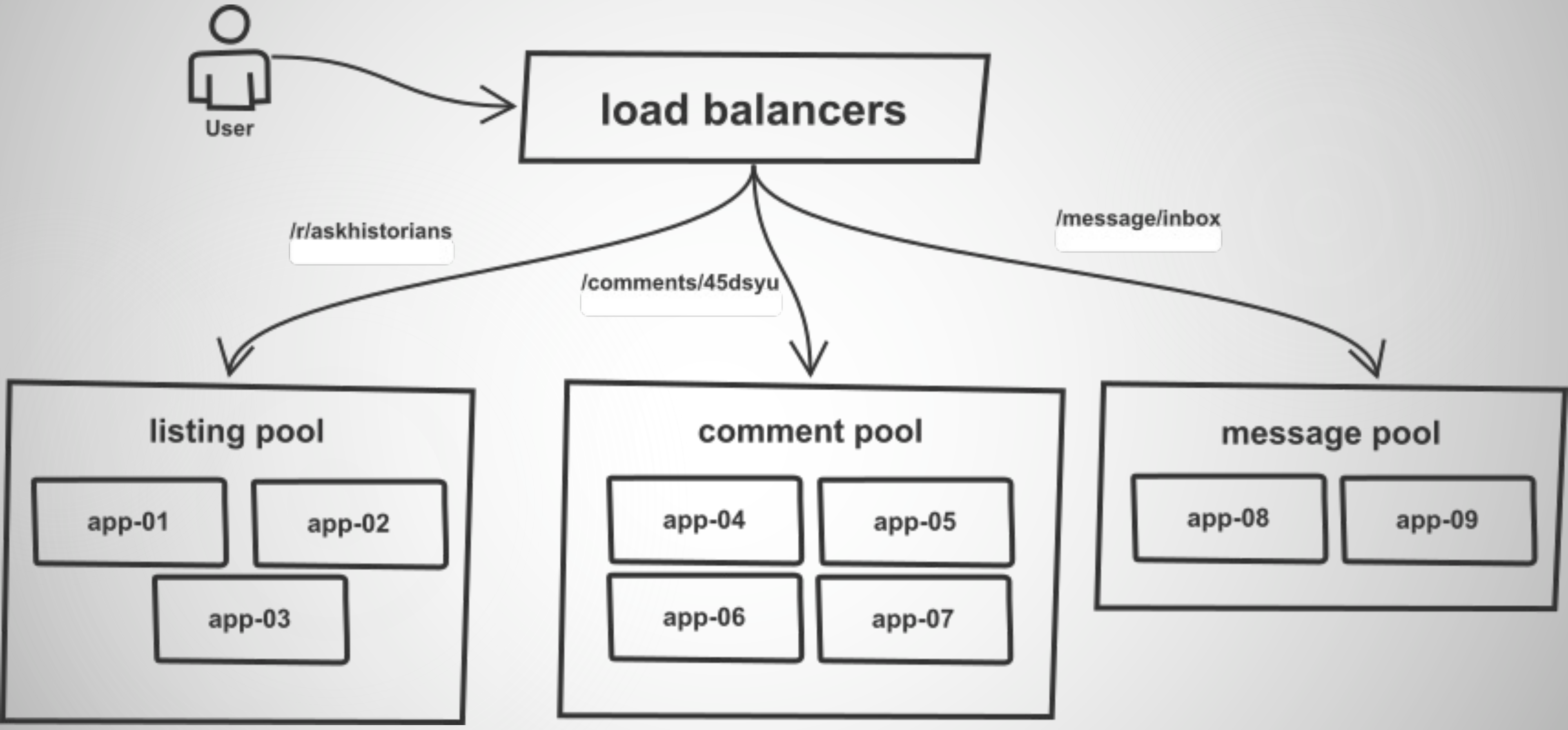


Source: https://automotiveembeddedsite.wordpress.com/why-autosar-what-it-is/

Netflix Architecture Diagram



Reddit Architecture Diagram



Source: https://redditblog.com/2017/06/02/the-evolution-of-code-deploys-at-reddit/

More real world architectures

- http://aosabook.org/en/index.html

Today's lecture – **How do we structure the software?**

- Introduction
- Defining software architecture
- Architecture in action
- Software architecture's elements
- Architectural erosion

Software architecture's elements

- A software architecture consists of
  - **Components**
  - **Connectors**
- Components and connectors are arranged into **configurations**

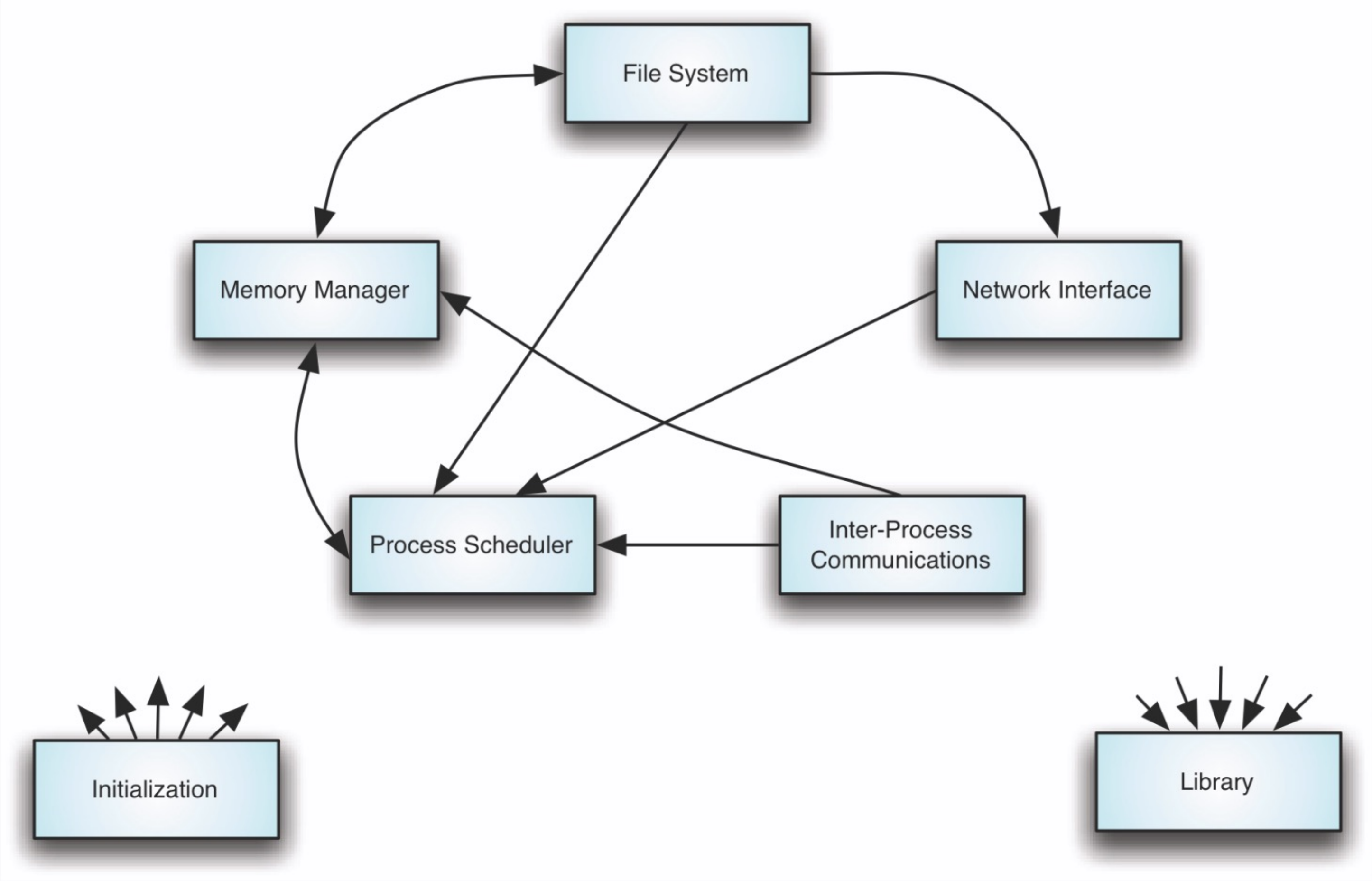
Today's lecture – **How do we structure the software?**

- Introduction
- Defining software architecture
- Architecture in action
- Software architecture's elements
- Architectural erosion

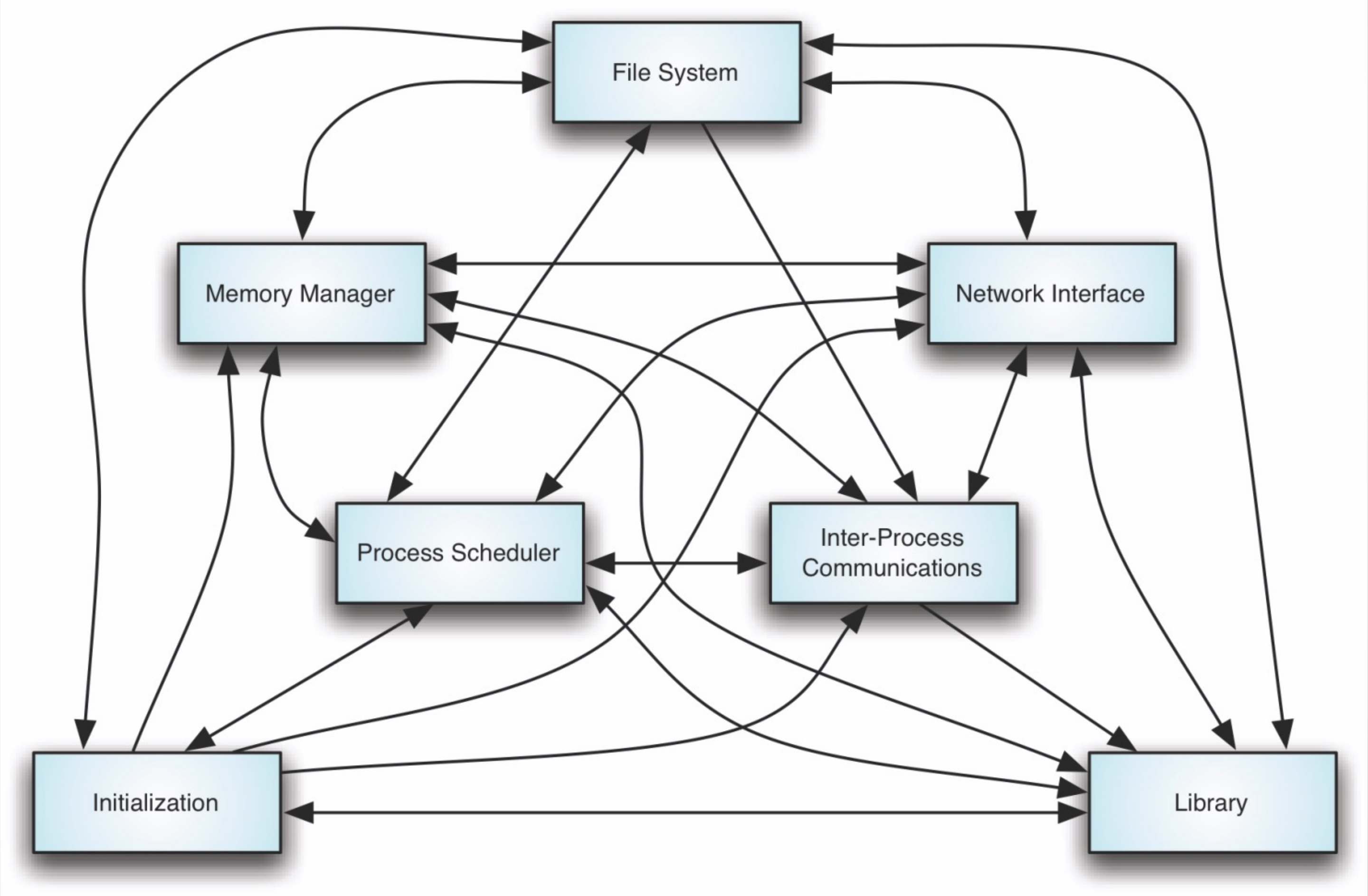
Prescriptive vs. descriptive architecture

- *Prescriptive architecture*
  - *as-designed/as-intended* architecture
- *Descriptive architecture*
  - *as-implemented/as-realized* architecture

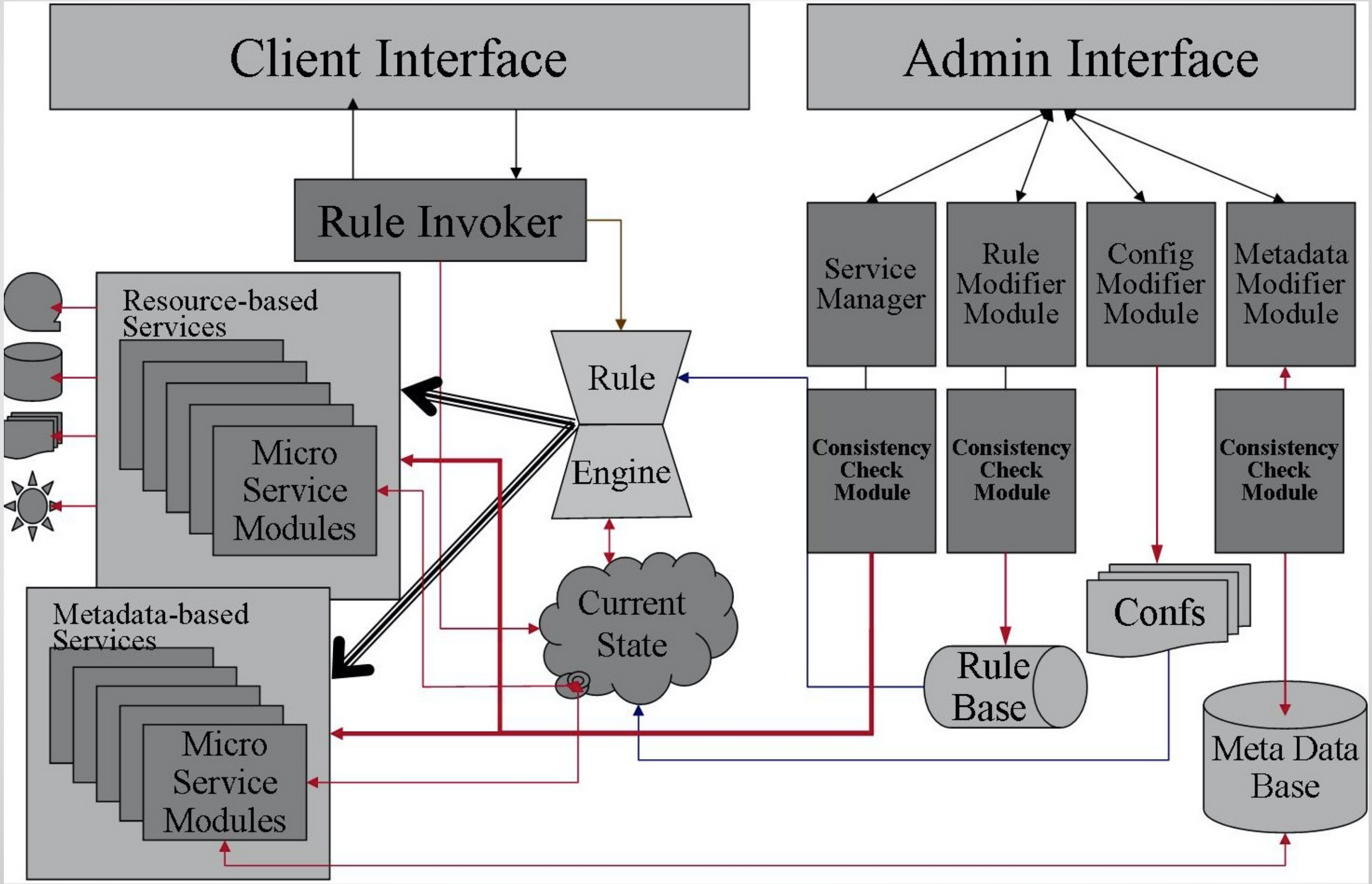
Linux – prescriptive architecture



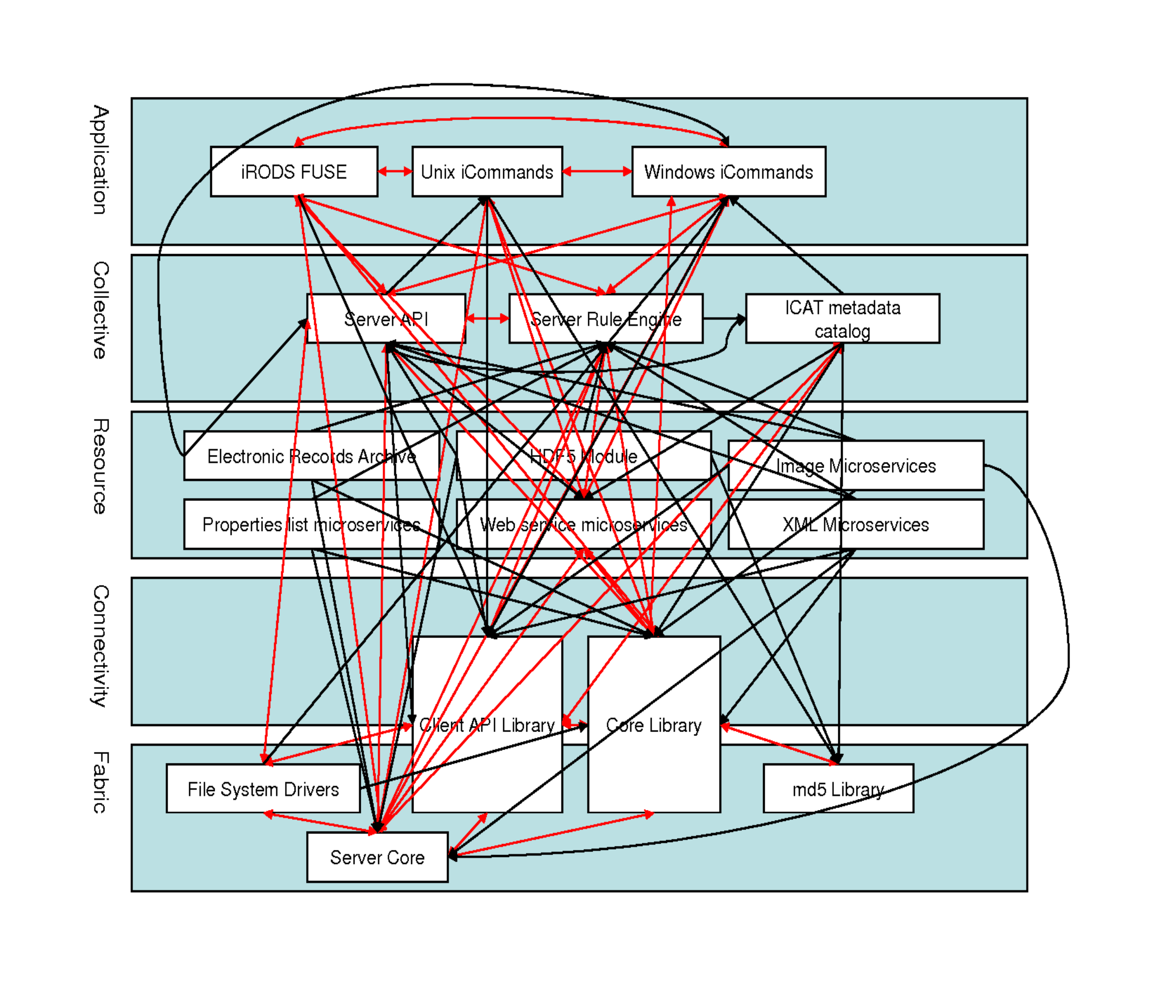
Linux – descriptive architecture



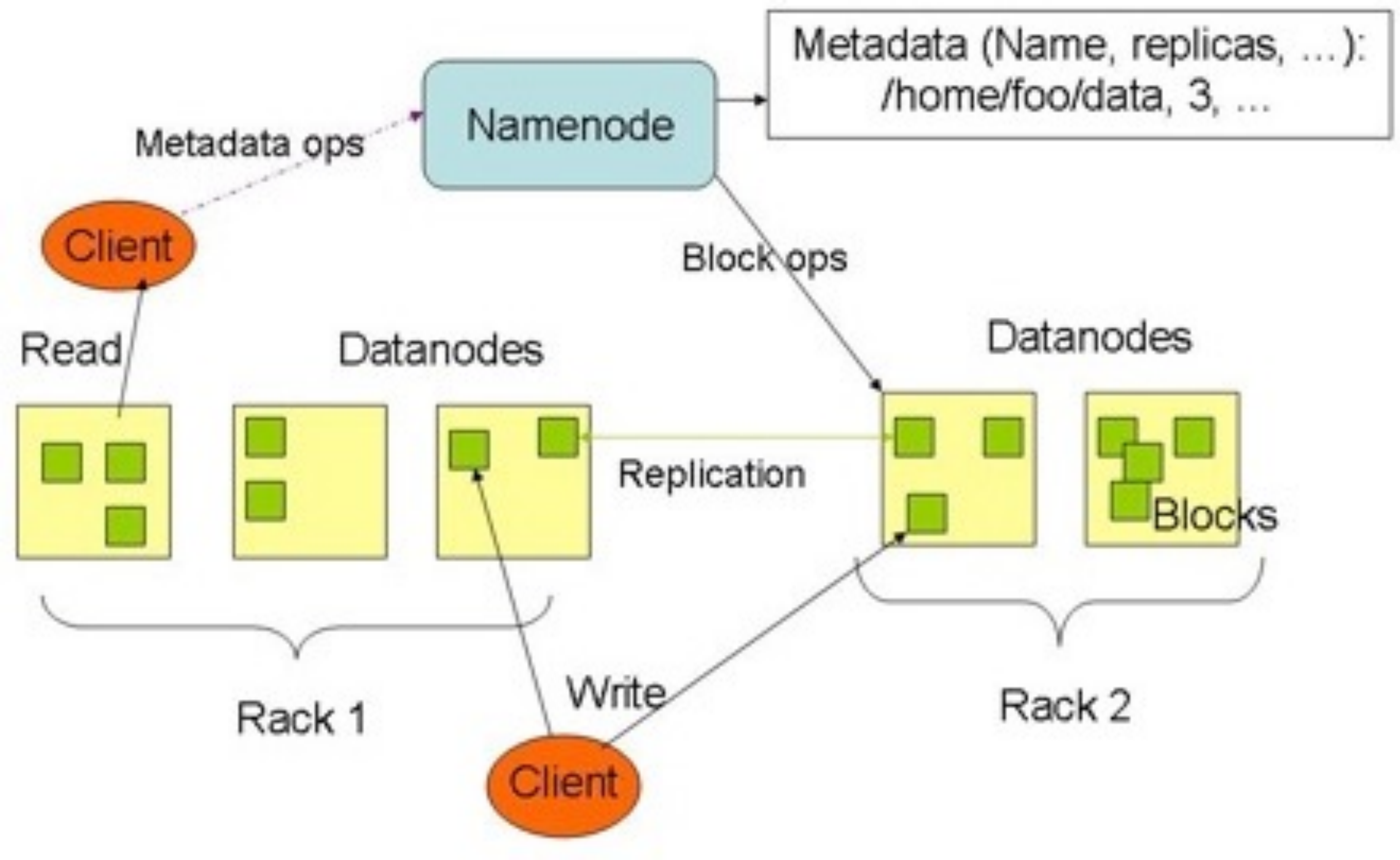
iRODS – prescriptive architecture



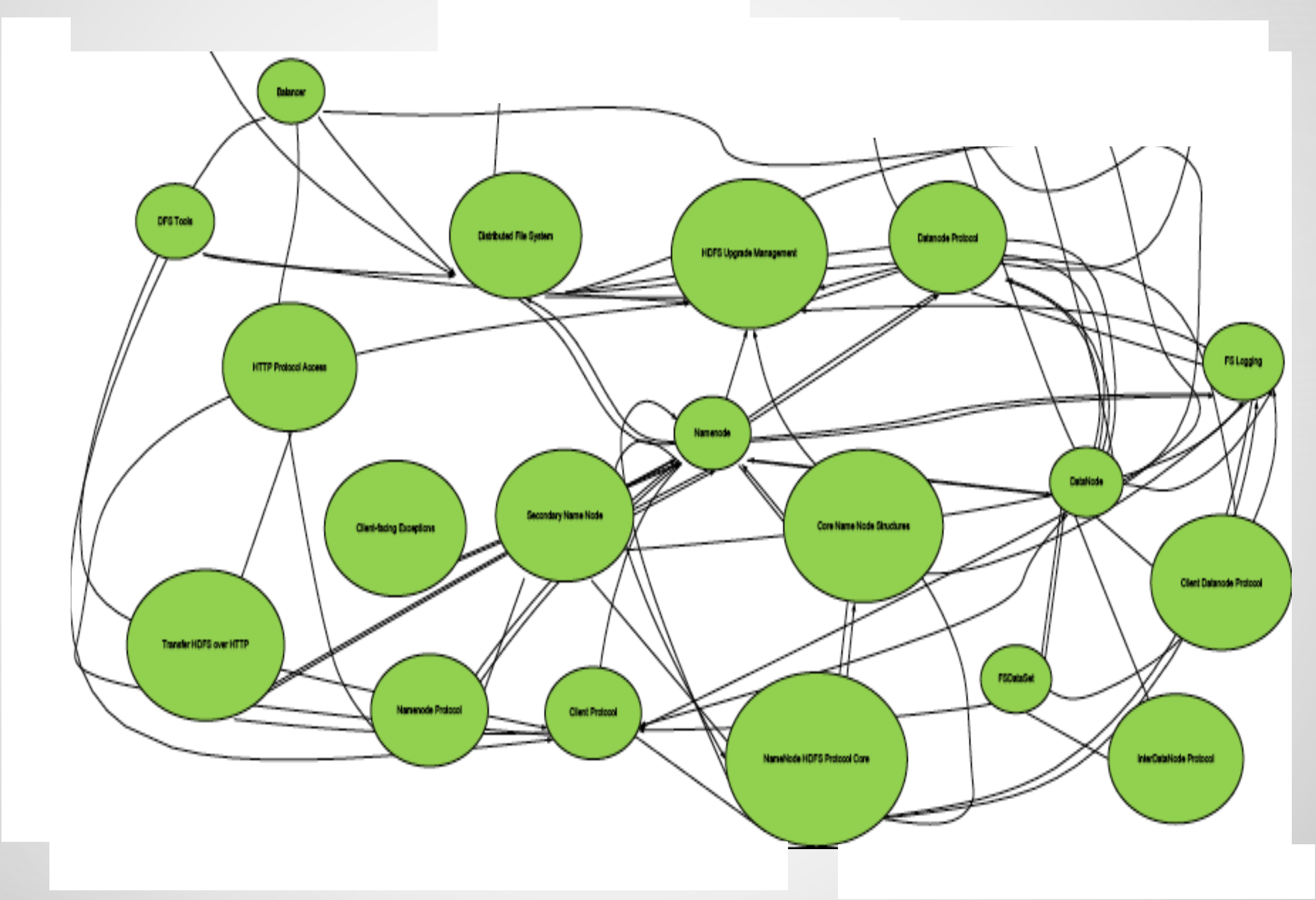
iRODS – descriptive architecture



HADOOP – prescriptive architecture



HADOOP – descriptive architecture



Architectural erosion

- When a system evolves, ideally its prescriptive architecture is modified first
- In practice, the system – and thus its descriptive architecture – is often directly modified

Architectural erosion – why does it happen?

- developer sloppiness
- short deadlines
- lack of (documented) prescriptive architecture
- code optimizations
- inadequate techniques or tool support

Summary

- Software architecture is a very powerful tool in software engineering
  - intellectual control, conceptual integrity, communication, reusability, maintainability
- A software system's architecture is:
  - the set of *principal design decisions* about the system
  - the blueprint for its construction and evolution
- A software architecture consists of components and connectors, arranged into configurations
- Architectural erosion happens when a system evolves and its prescriptive architecture is not updated

Next time

- More on software architecture