Informatics 43

LECTURE 8

"HOW DO WE STRUCTURE THE SOFTWARE (PART 2)"

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

- Software architecture - recap
- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

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

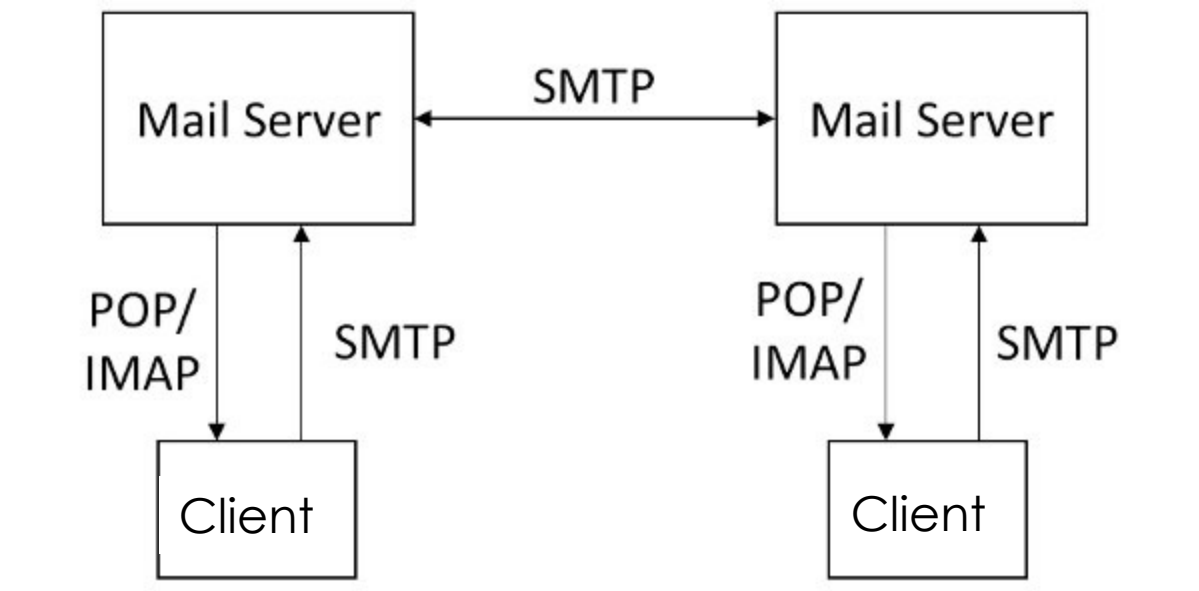
Today's lecture – **How do we structure the software?**

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

Email: functional specification

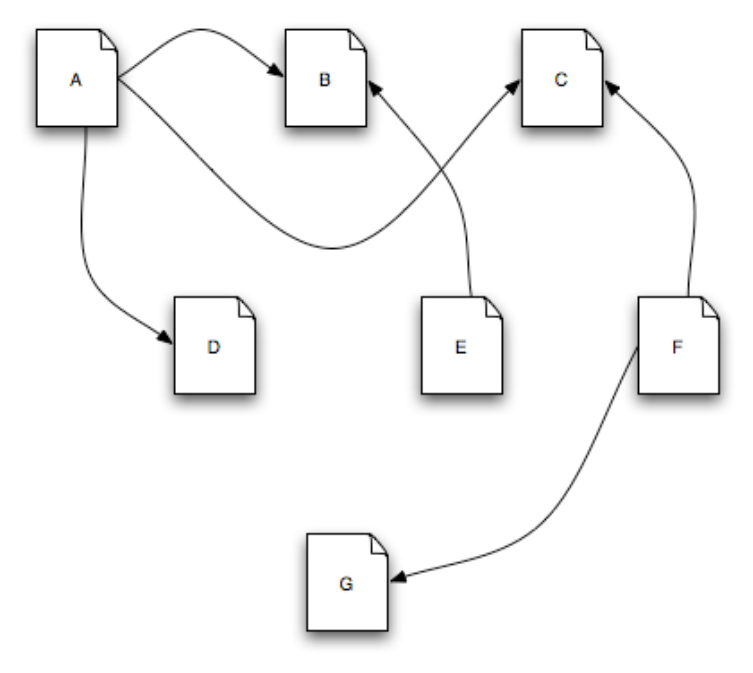
- Send and receive messages
- Get your own messages, not others'
  - Addressing scheme
- Store messages
- Electronic
- Blocks spam

Email - Architecture



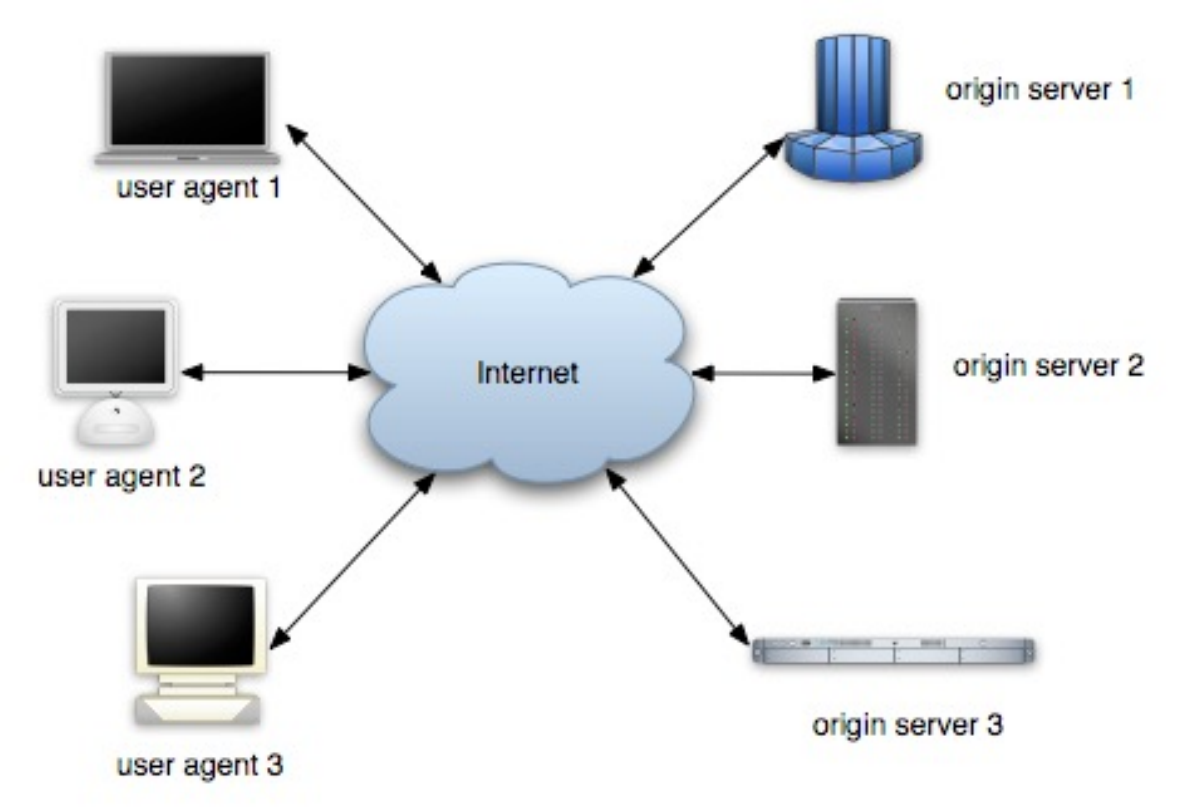
Architecture in action: WWW

This is one way to describe the Web



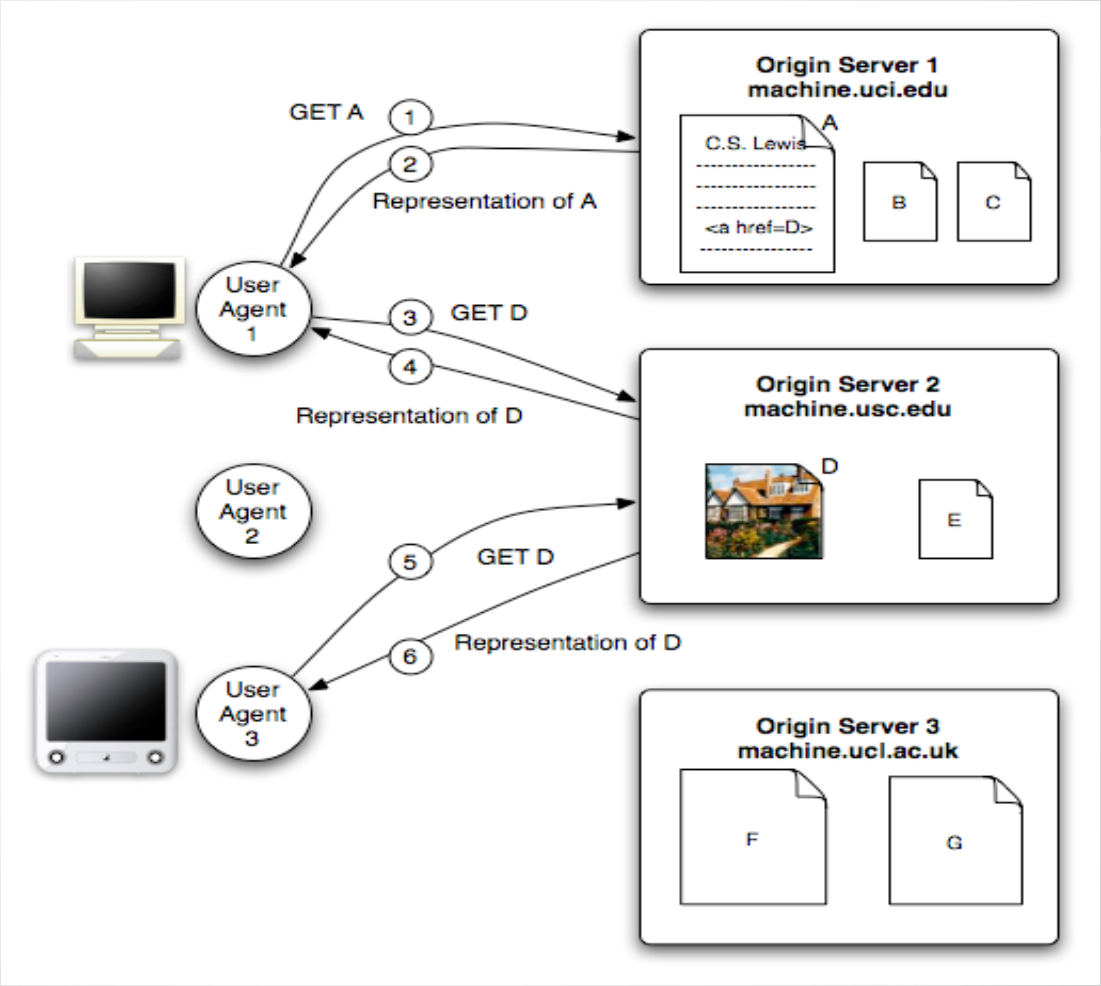
Architecture in action: WWW

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

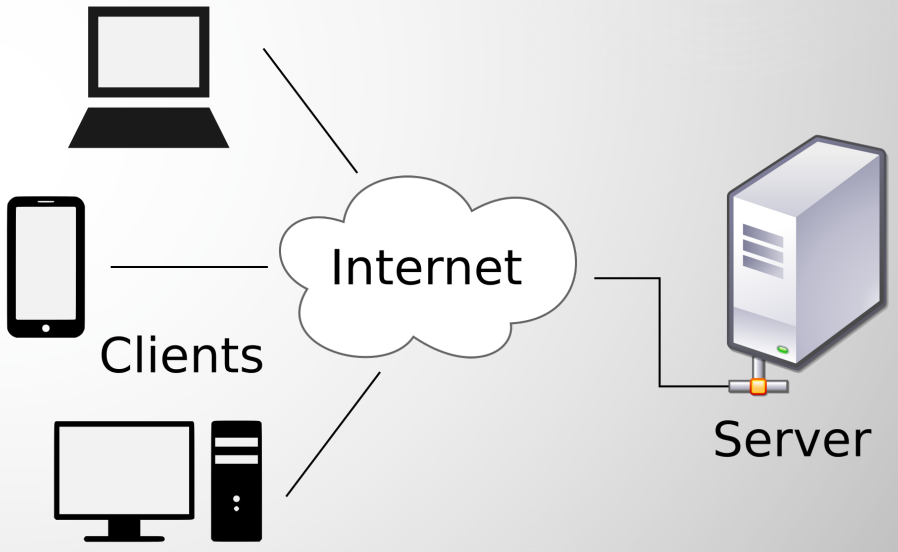
- Functional requirements?
  - Friend lists
  - Status updates
  - News feed
  - Comments/likes/reactions
  - Messaging
  - Photos
  - Location services
  - 3rd party apps
  - …

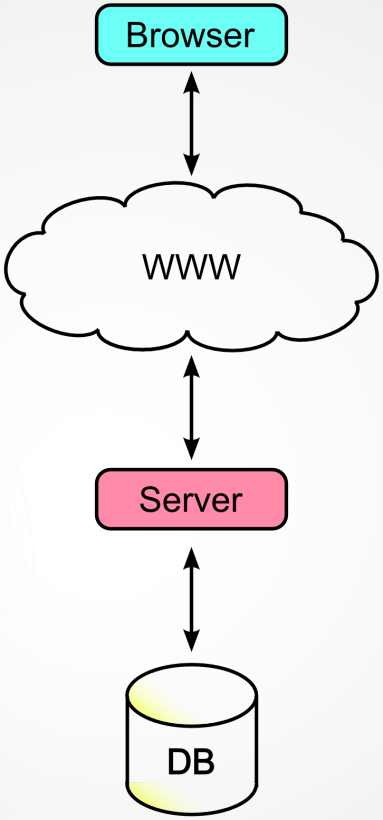
Facebook Architecture

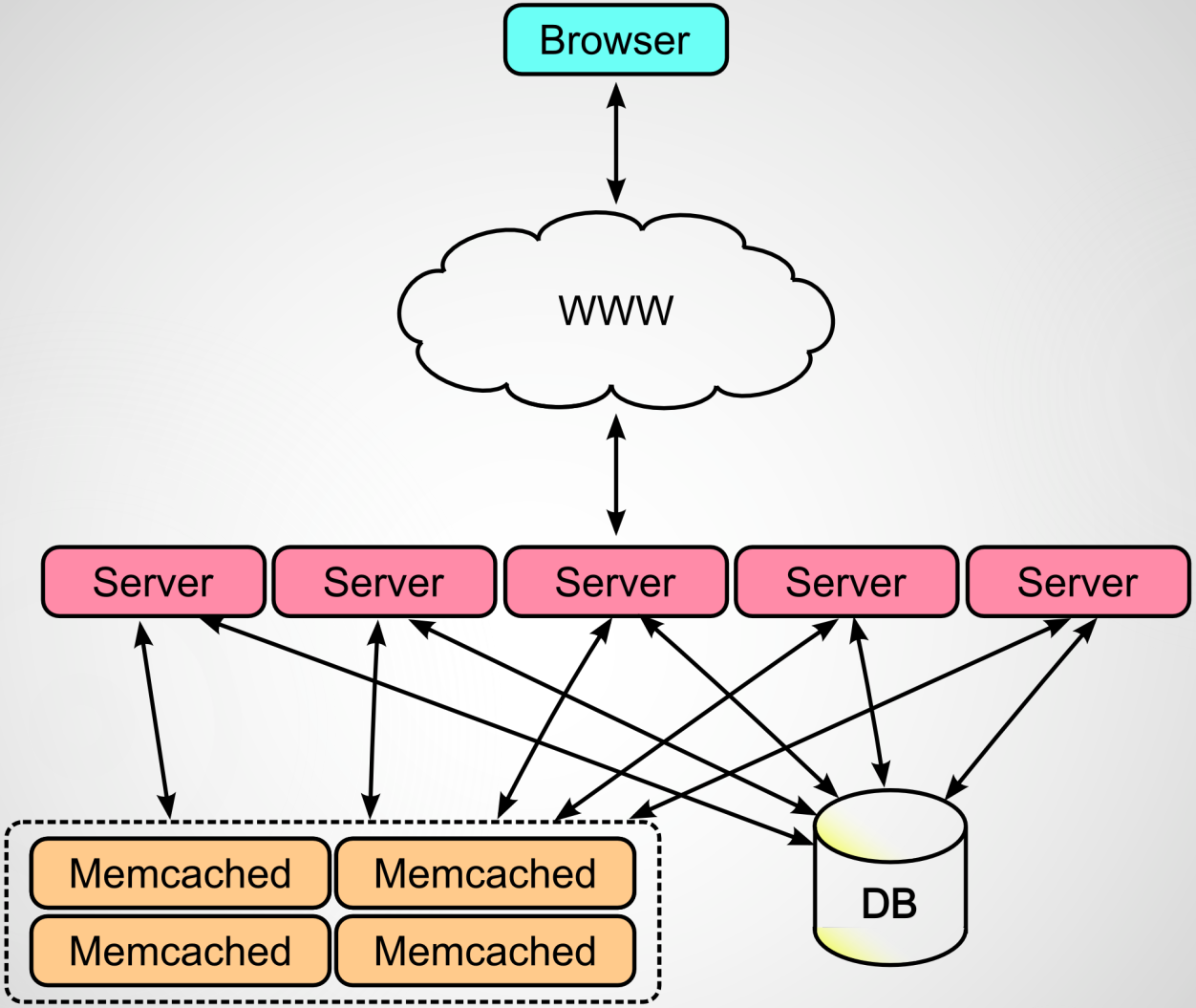
- Non-functional requirements?
  - Efficiency/performance
  - Scalability
  - Availability
  - Security
  - Privacy
  - Reliability
  - Portability

Client Server Architecture

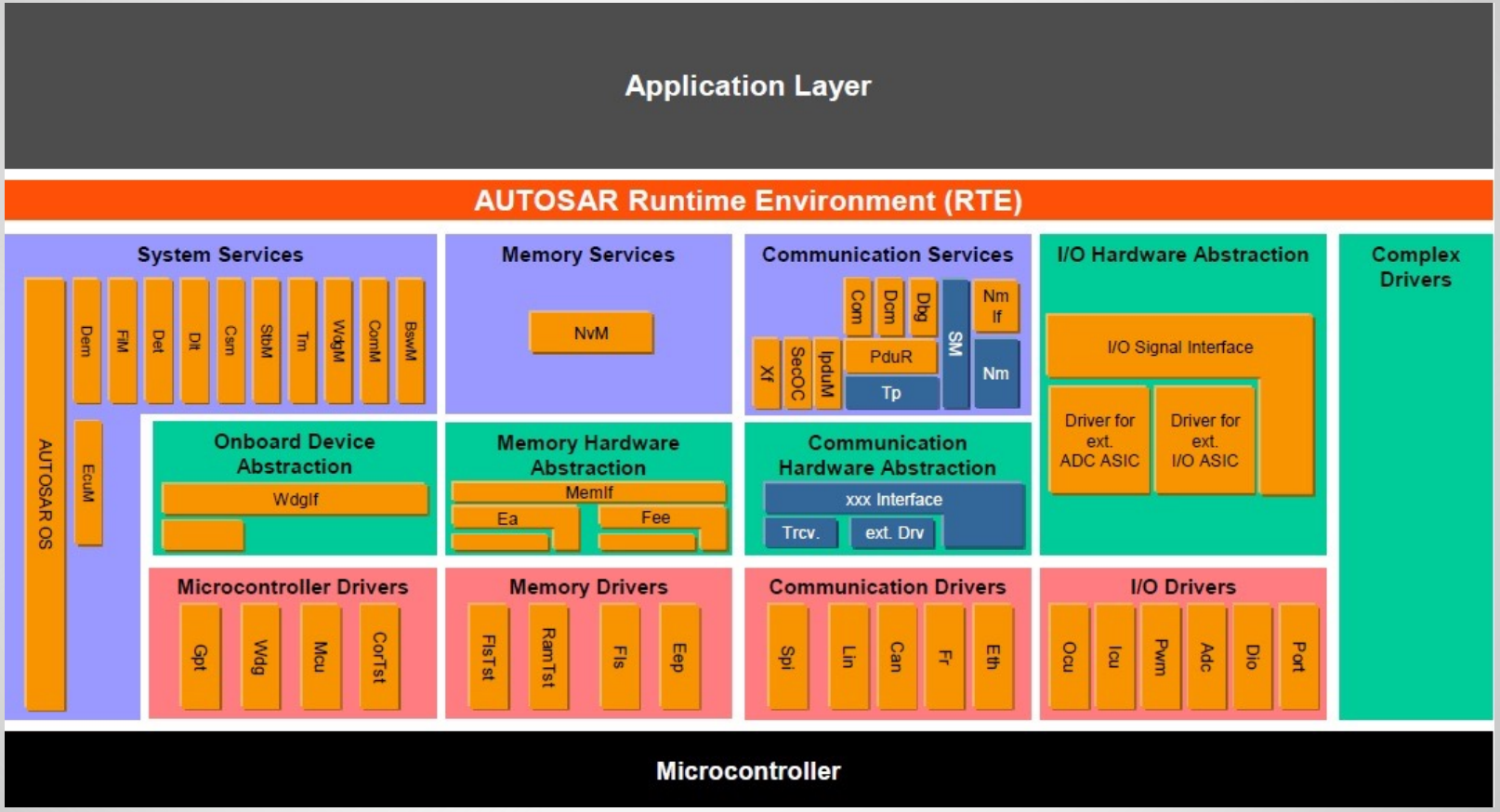
- Most common distributed system architecture
- Clients make requests of servers
- Centralized
  - data
  - security
  - access




Browser
WWW
Server
DB

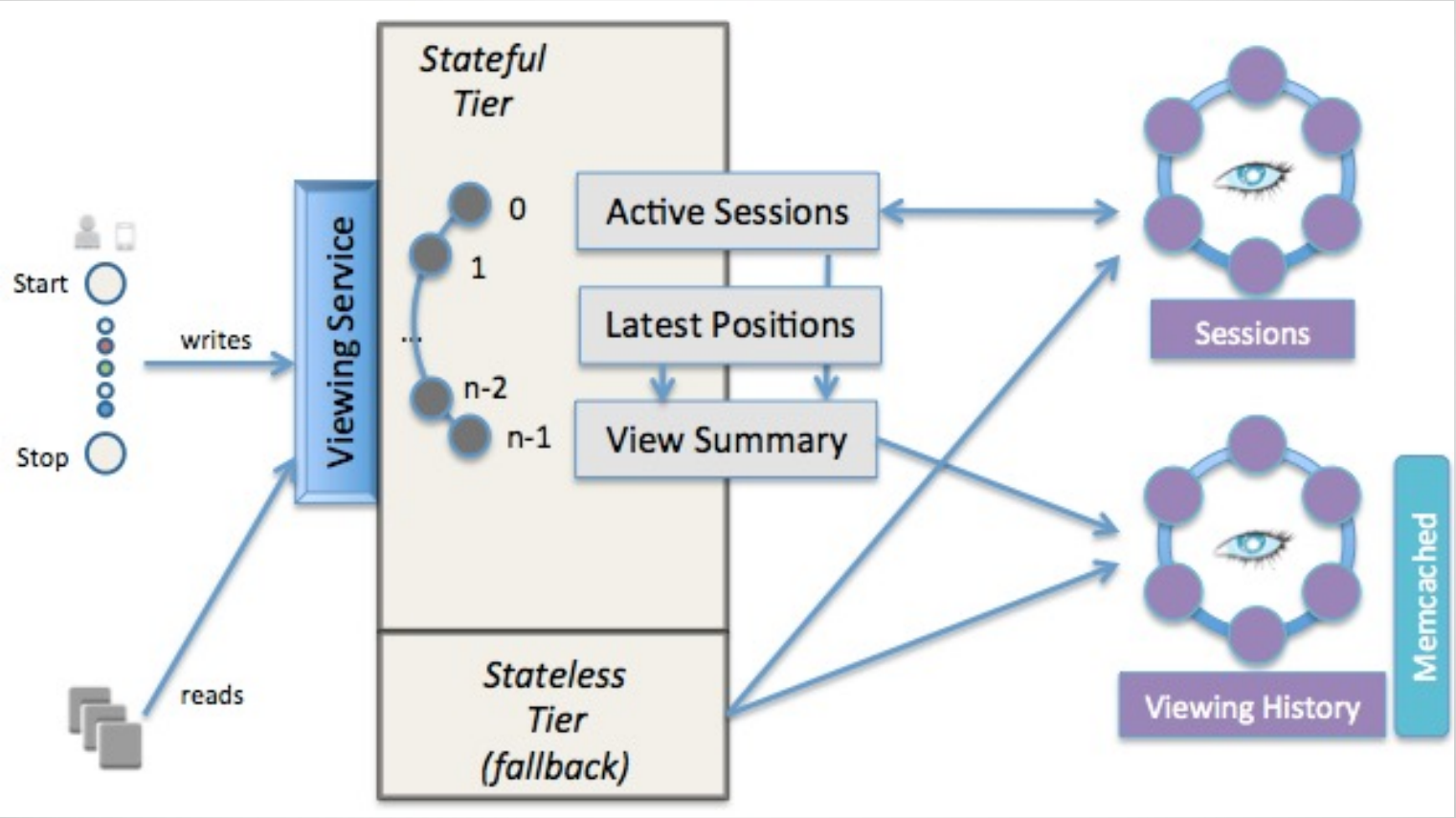

Browser
WWW
Server
Server
Server
Server
Server
Memcached
Memcached
Memcached
Memcached
DB

AUTOSAR Architecture Diagram

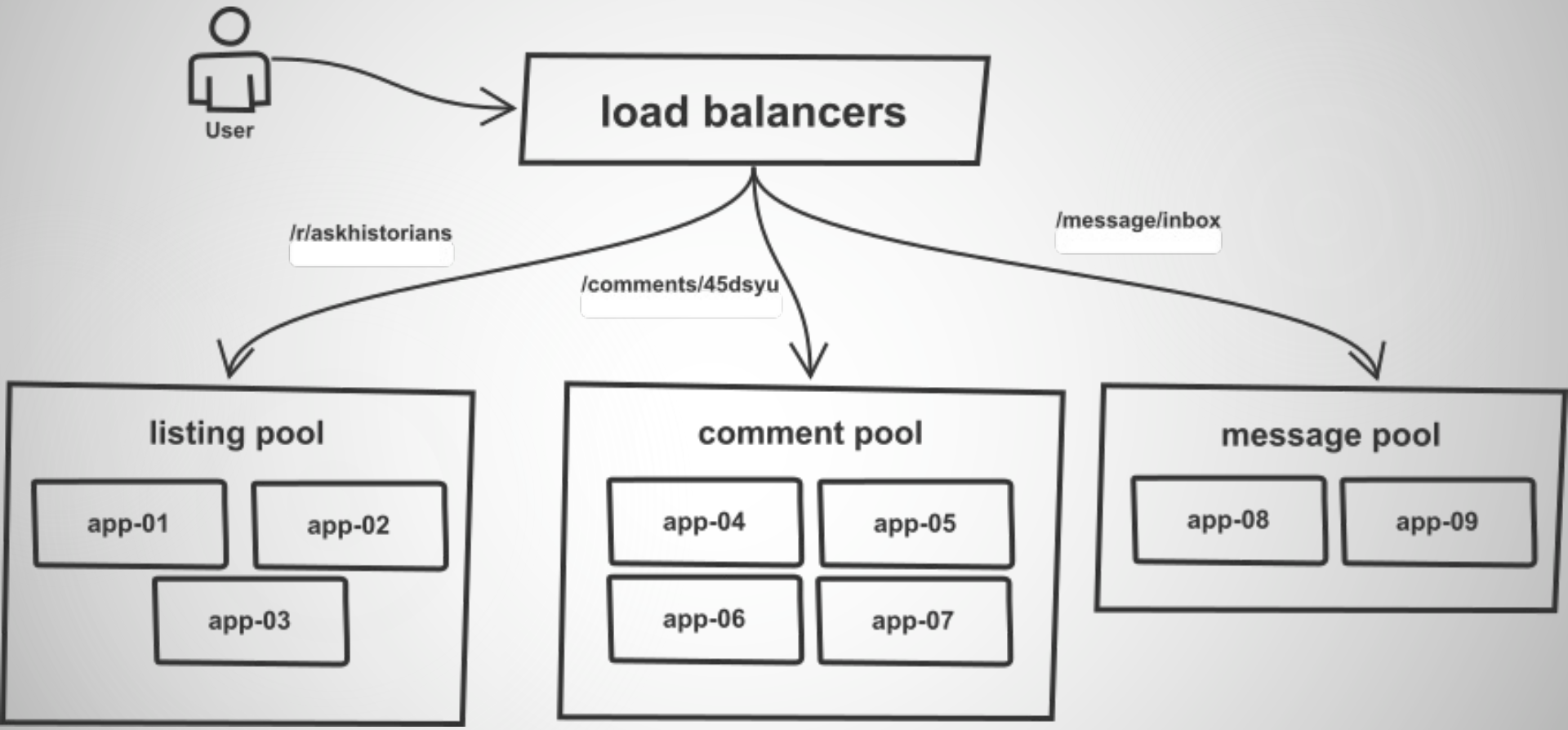


Source: https://automotiveembeddedsite.wordpress.com/why-autosar-what-it-is/

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**

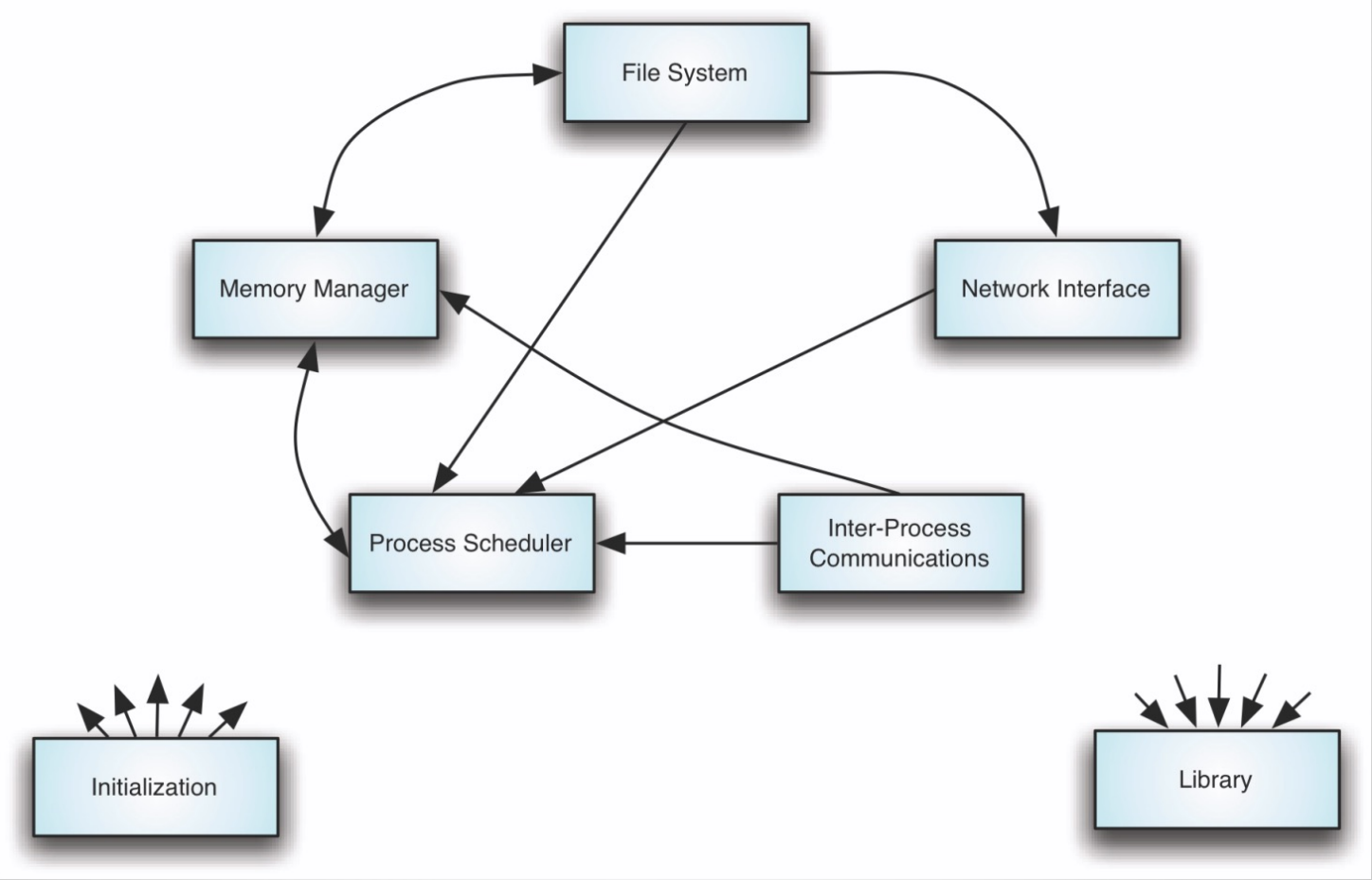
Today's lecture – **How do we structure the software?**

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

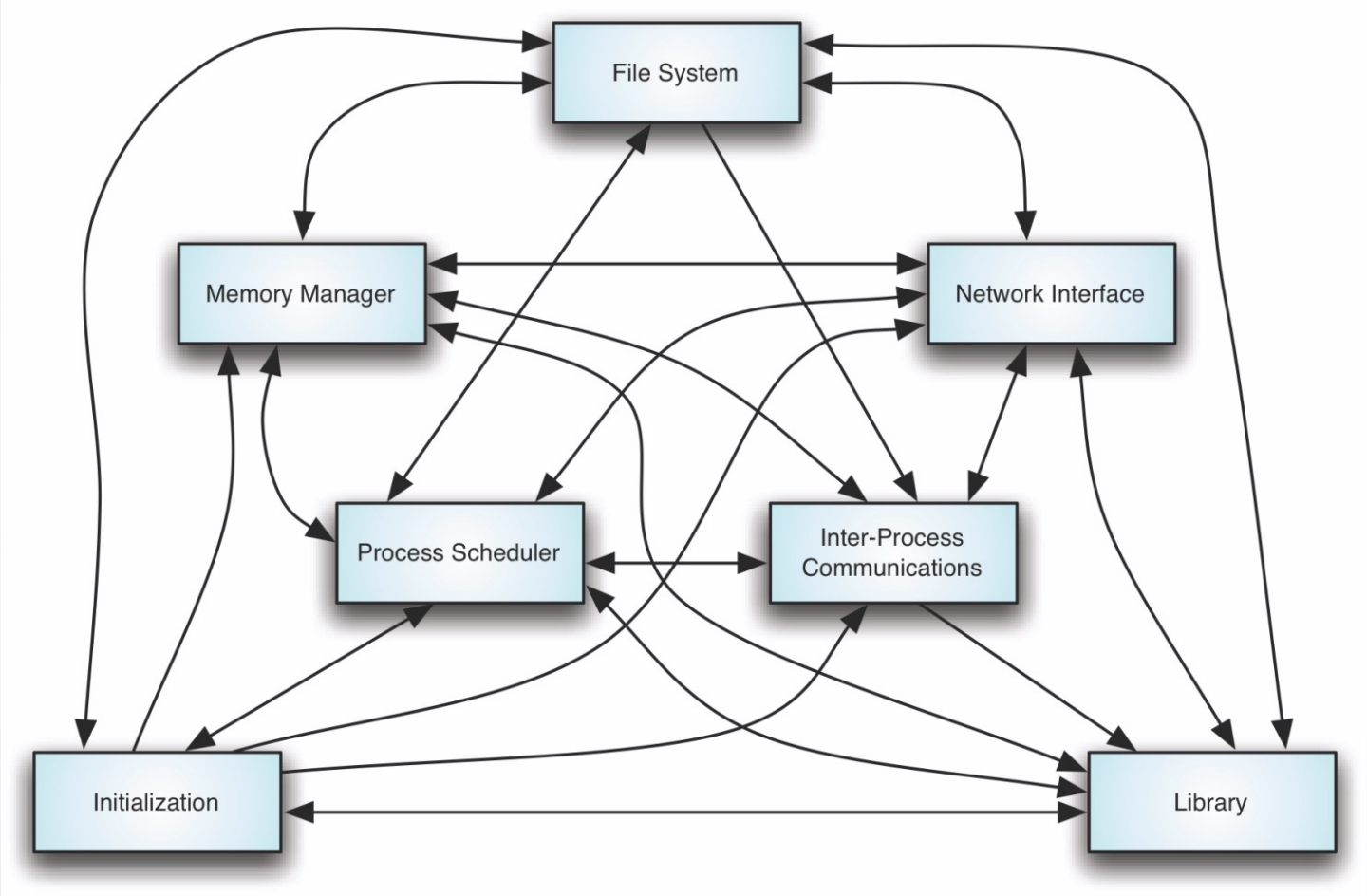
Prescriptive vs. descriptive architecture

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

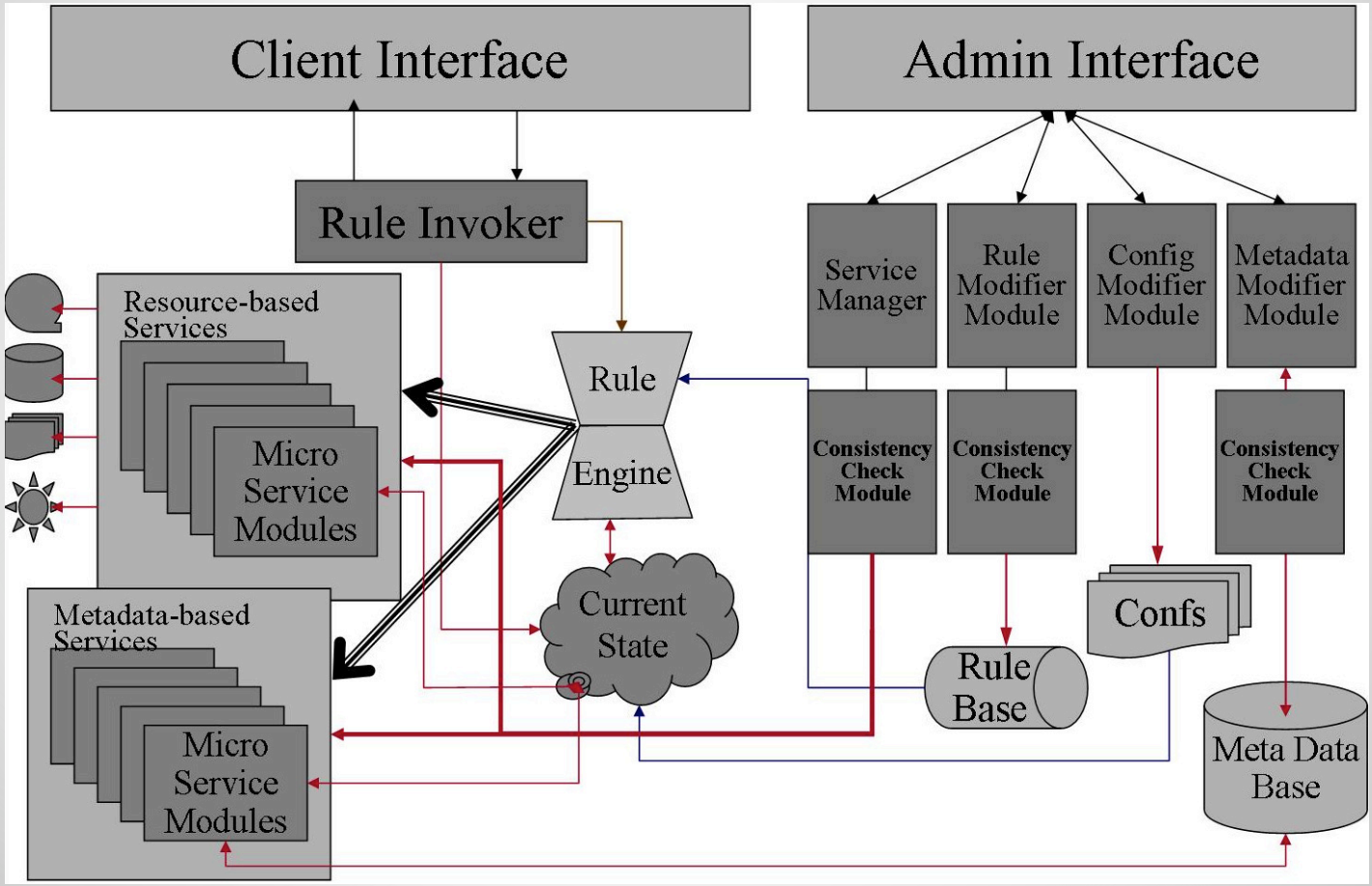
Linux – prescriptive architecture



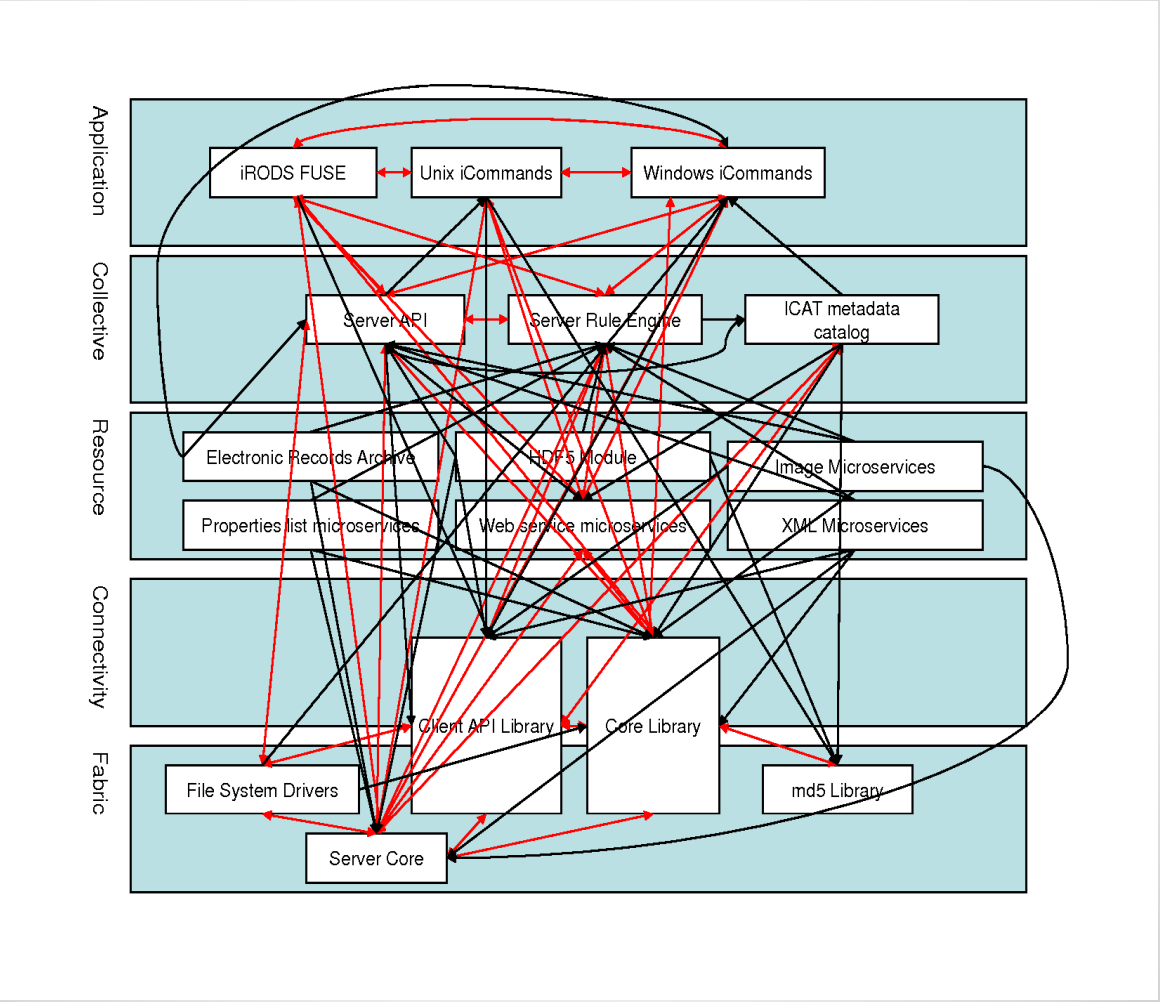
Linux – descriptive architecture



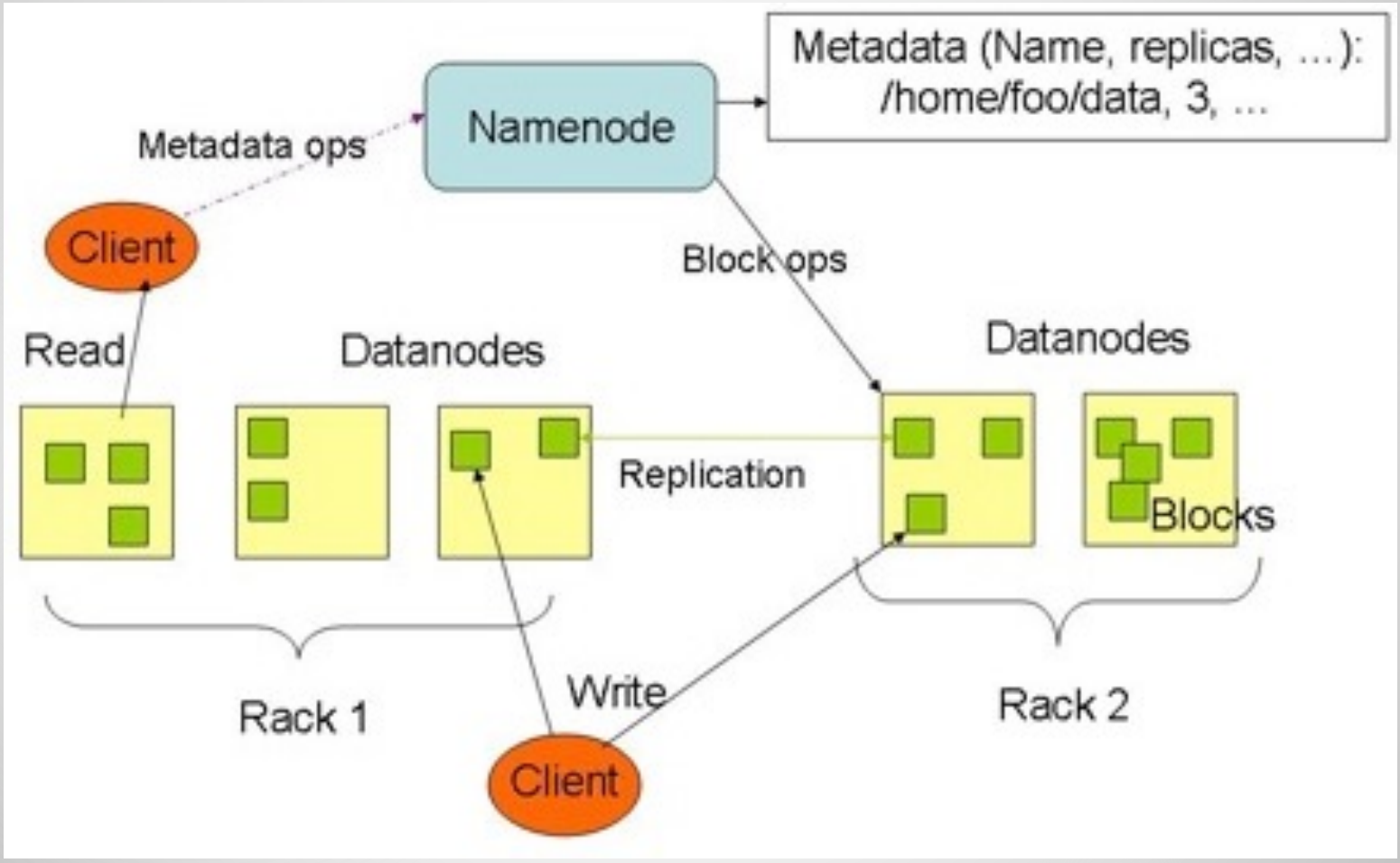
iRODS – prescriptive architecture



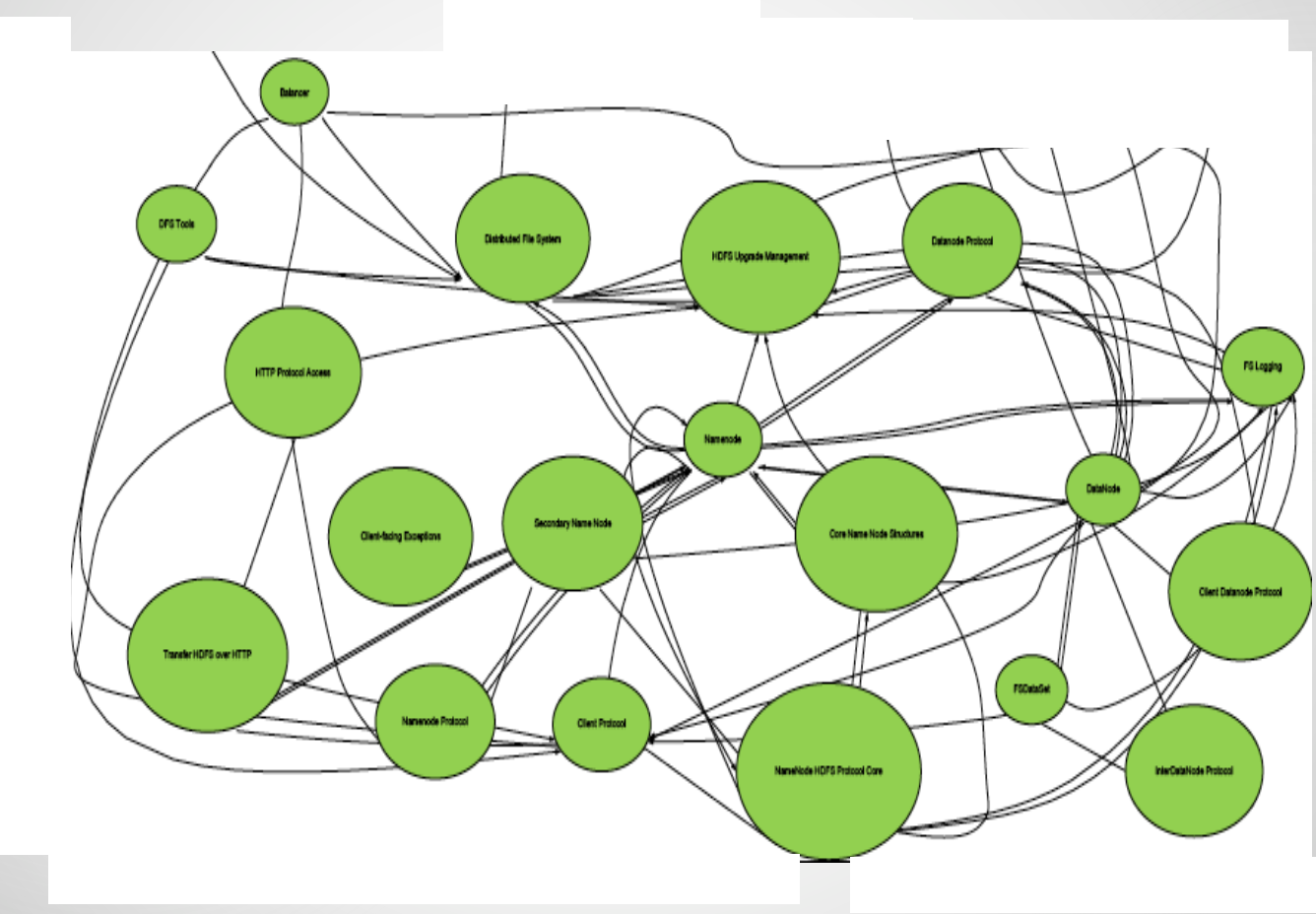
iRODS – descriptive architecture



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

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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

Architectural style

Contemporary style:



Arts and Crafts style:


Software architectural style

- A named collection of architectural design decisions that
  - are applicable in a given development context
  - constrain architectural design decisions
  - result in beneficial qualities in each resulting system
- A named, commonly used set of components/connectors/configurations
  - Each component/connector has its own job

Today's lecture

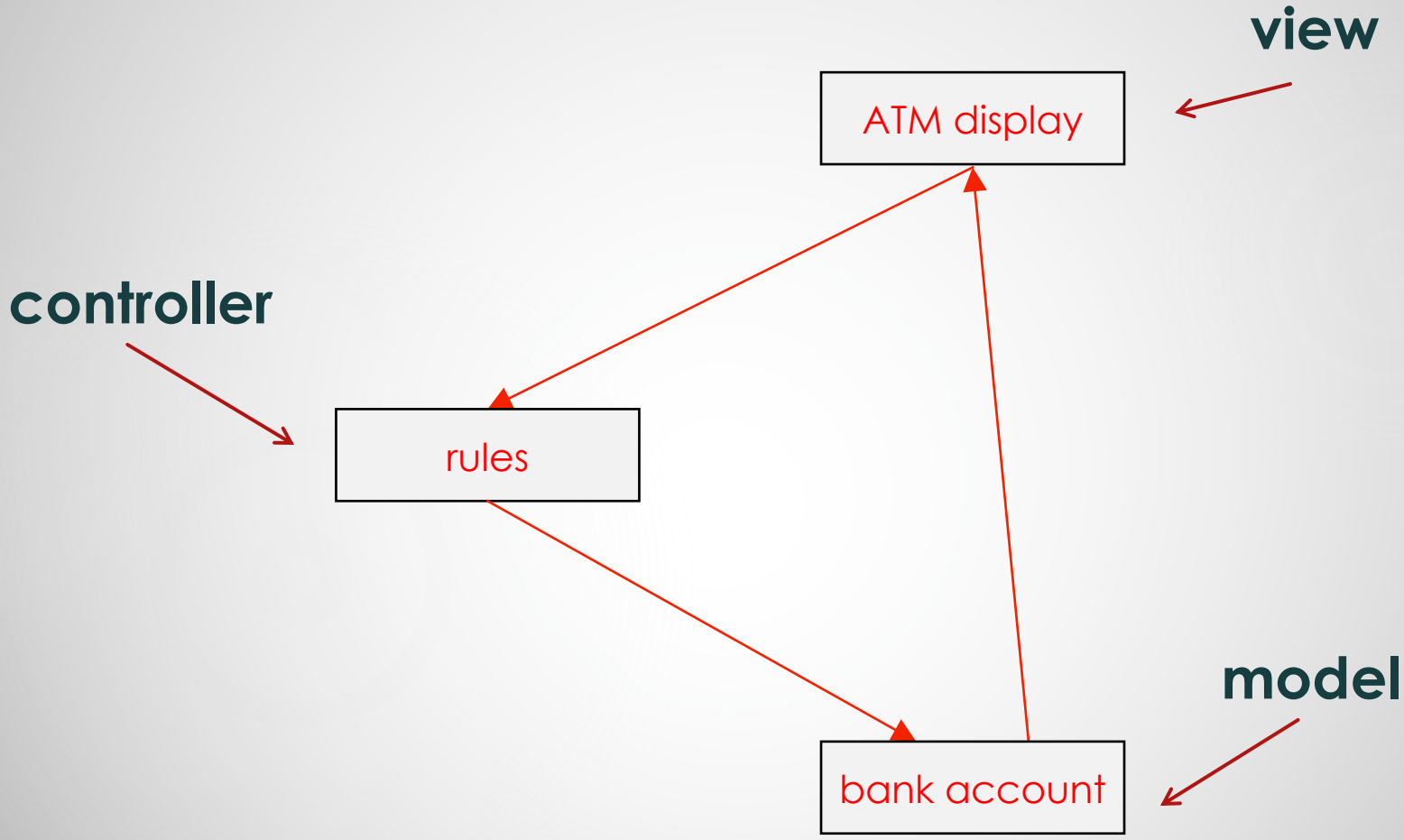
- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

Example 2: ATM

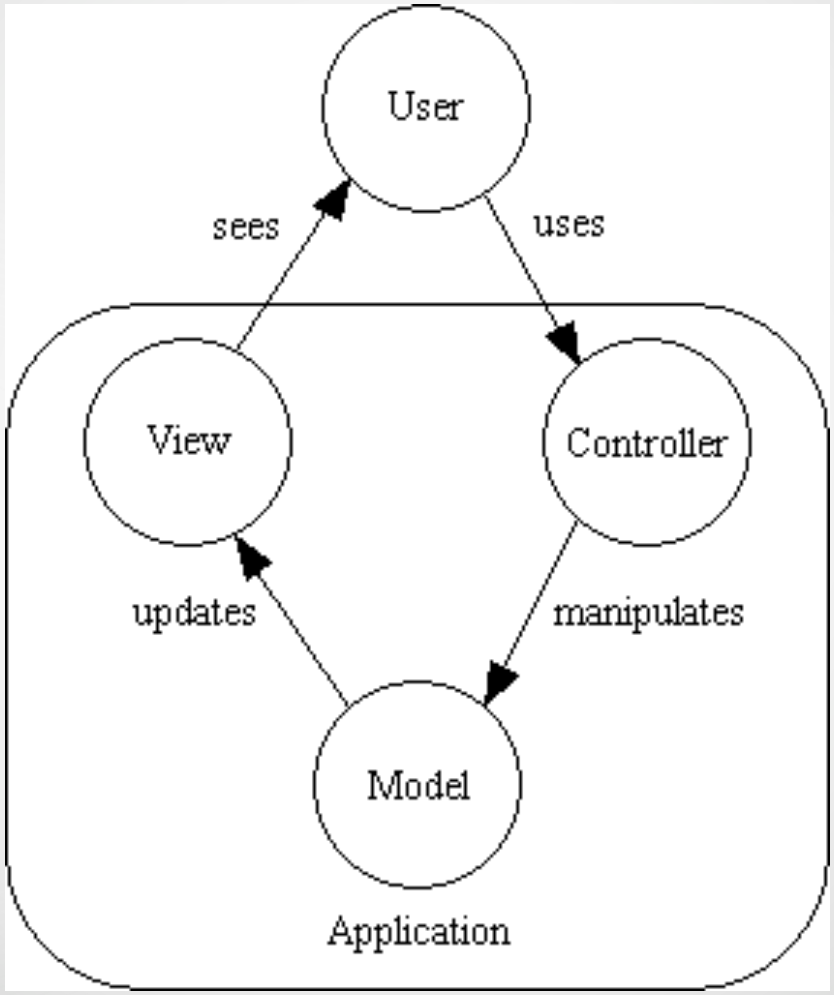
Goals:

- Withdrawals, deposits, transfers, balance checks
- Separation of concerns:
  - want to experiment with different UIs
  - isolate business rules (due to frequent anticipated changes)

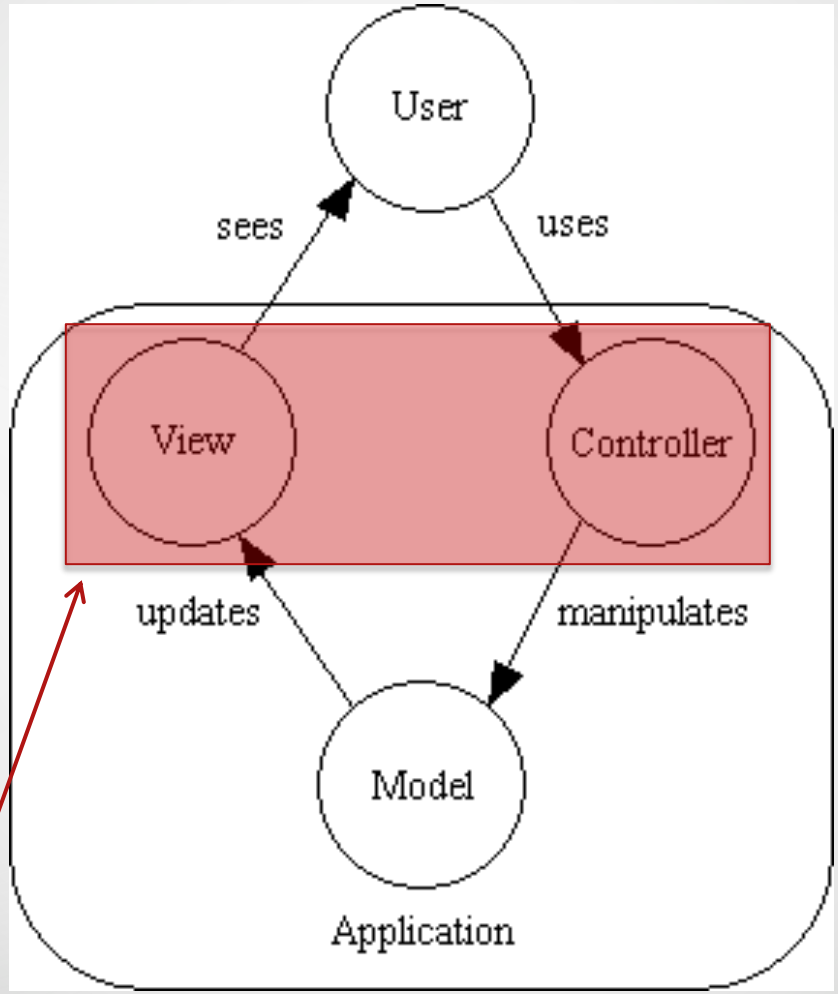
Style 2: model-view-controller



Style 2: model-view-controller



Style 2: model-view-controller



User interface

Style 2: model-view-controller

- Key benefits:
  - Modularity
  - Anticipation of change

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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

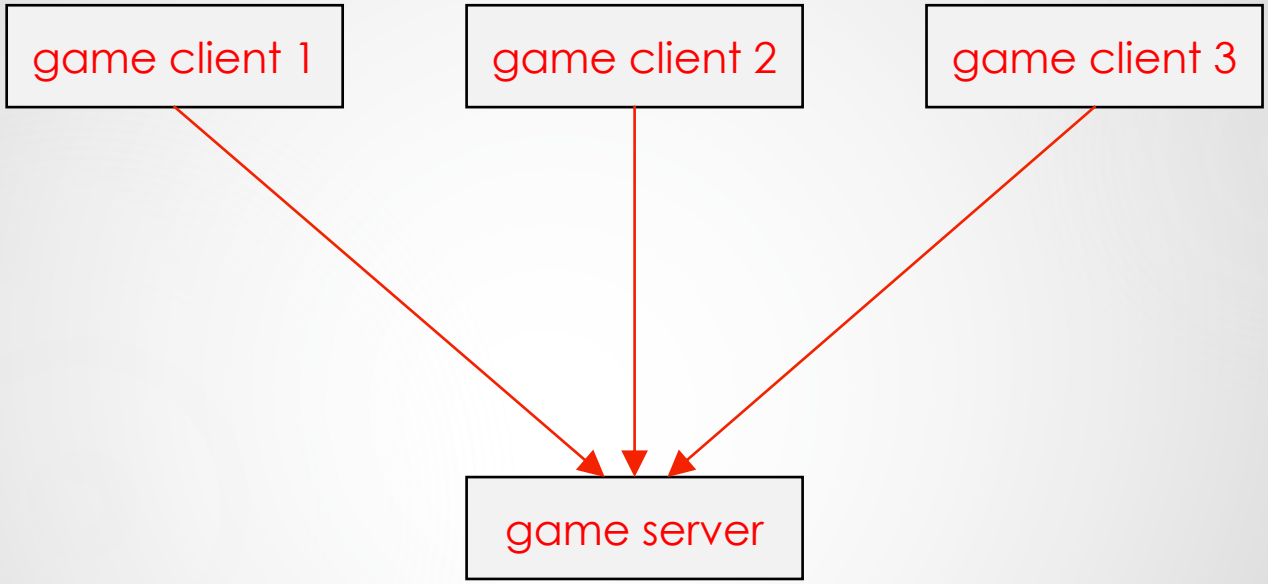
Example 3: Video Game

Source: https://www.techhnews.com/fortnite-for-android-epic-tells-us-why-it-wont-be-on-googles-play-store/

Goals:

- Multi-player, online
- Control access to data and players

Style 3: client-server



Style 3: client-server

- Key benefits
  - Sharing data and services over a wide range of clients
  - Centralized control over access, functionality, data

Cloud computing has changed client-server

- Most modern client-server applications run on servers in the cloud
  - Software developers no longer have to manage physical servers
  - Instead, they are managed by a third party
  - Many cloud providers also provide application services
- Started by Amazon Web Services (AWS)
- Benefits
  - Scalability
  - More computing power
  - Cost-effective?
- Drawbacks
  - Security/privacy?
  - May not be cost-effective

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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

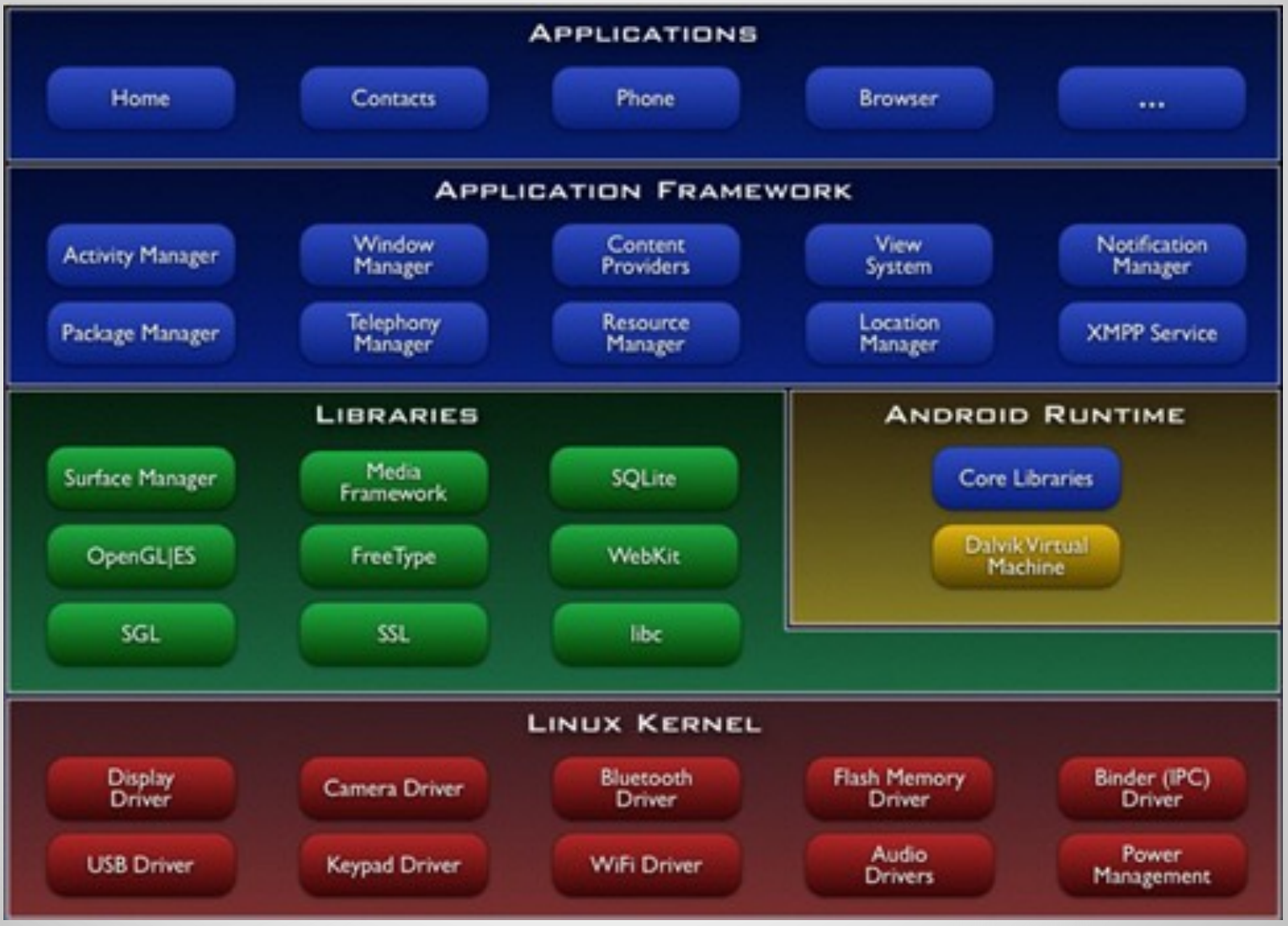
Example 4: Restaurant

Goals:

- Prepare and serve food to customers
- Enforce a specific protocol of interaction
  - abstract away irrelevant details
  - minimize the effects of changes
- Separate concerns: keep each part focused on a specific task

Style 4: layered

Style 4: layered (Android)



Source: http://www.zdnet.com/article/how-android-works-the-big-picture/

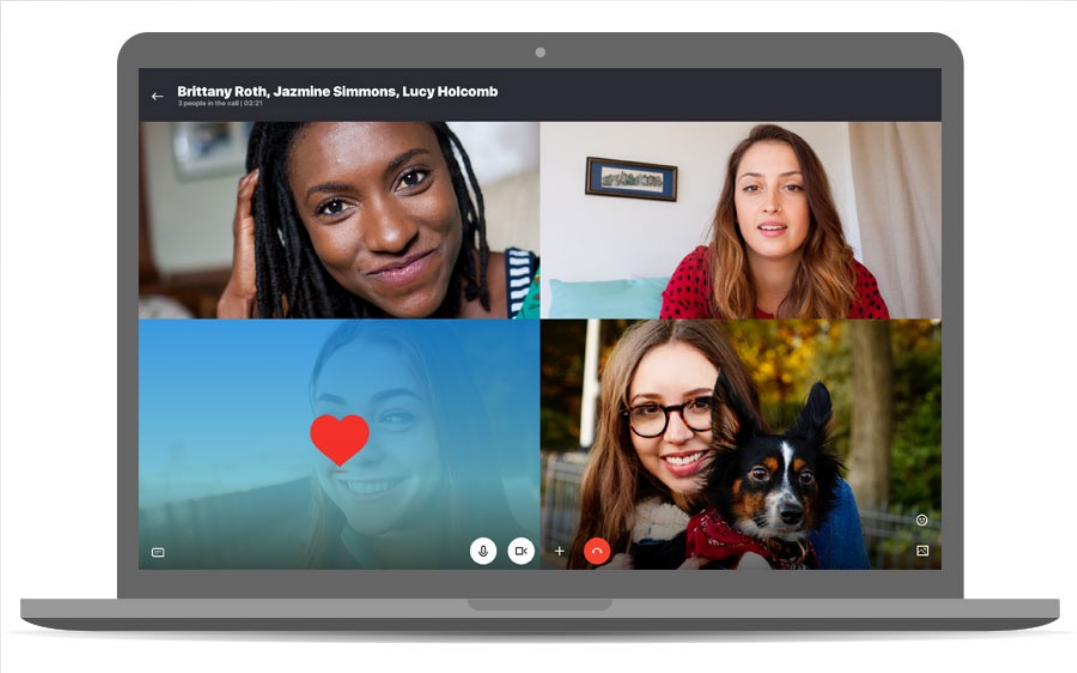
Style 4: layered

- Key benefits
  - Modularity
  - Abstraction
  - Anticipation of Change
  - Reuse

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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

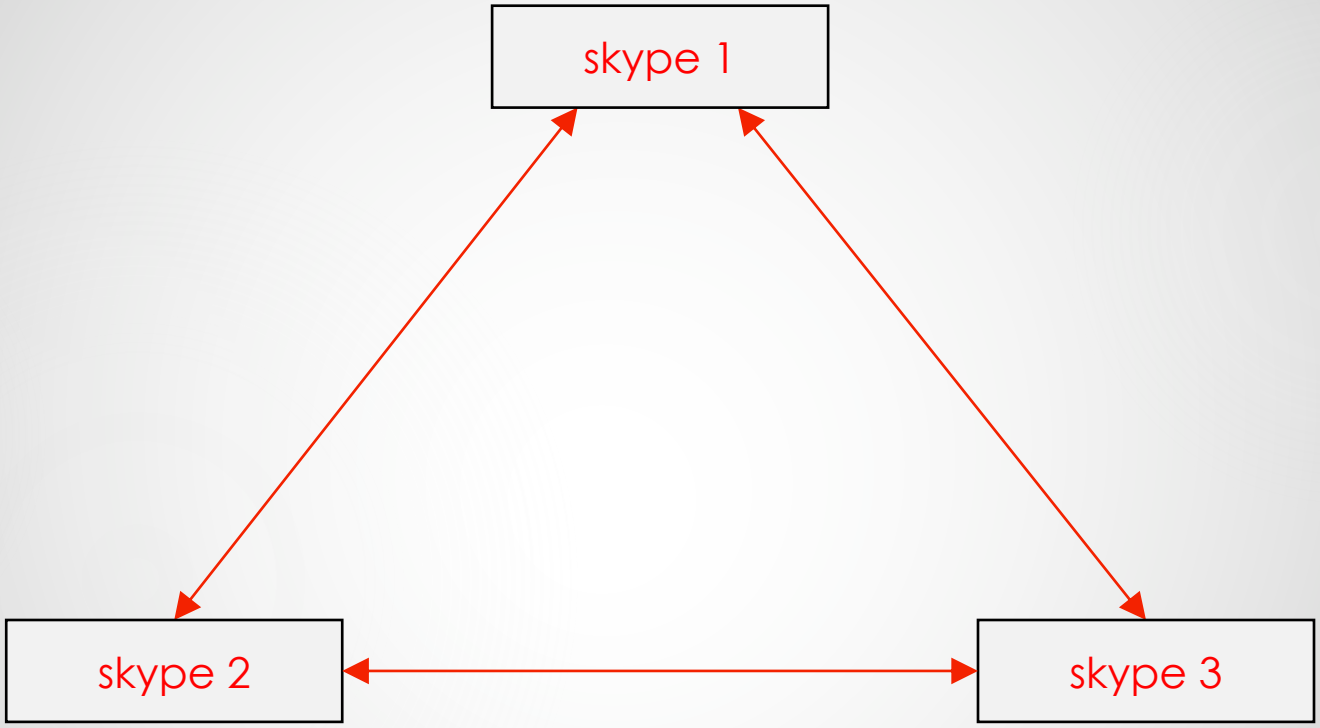
Example 5: Skype



Goals:

- Online (multi-person) video chat and voice calls
- Share same functionality between multiple people without slowing down video/audio streaming

Style 5: peer-to-peer



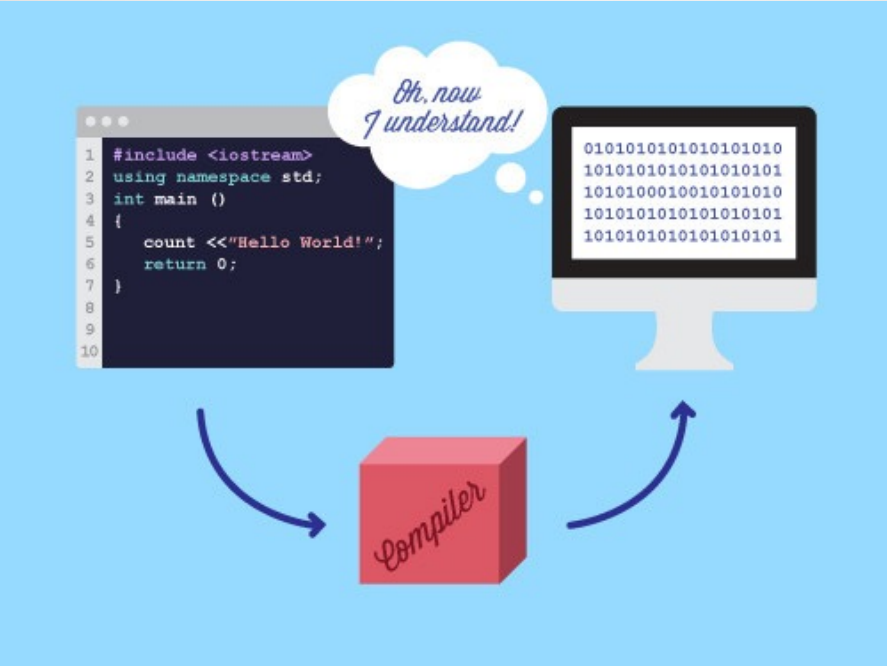
Style 5: peer-to-peer

- Key benefits
  - Efficient
  - Robust

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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

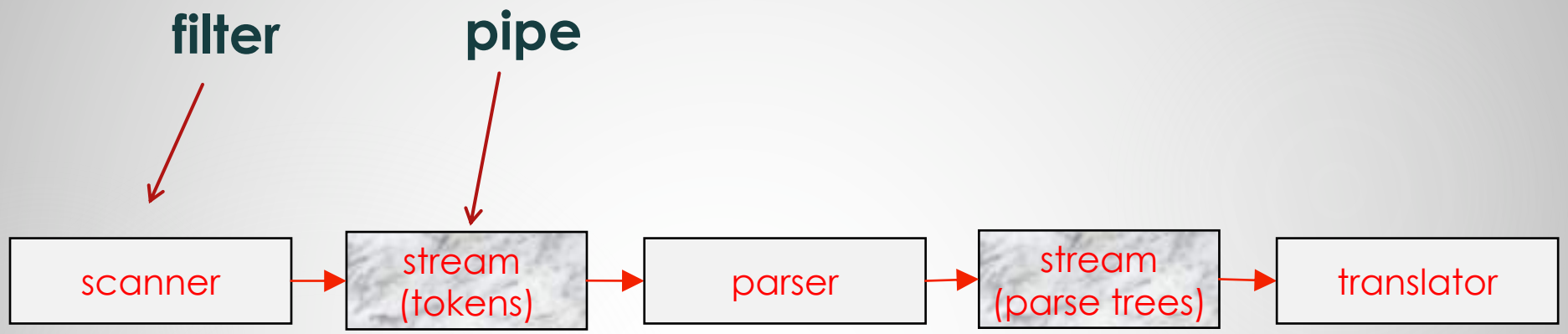
Example 6: Compiler

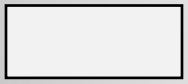


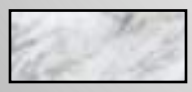
Goals:

- Translate a high-level programming language into a low-level one that computer can execute
- Separate different steps of translation so I can add/edit/remove/replace steps easily

Style 6: pipe-and-filter




component


connector

Style 6: pipe-and-filter


```
ls -l | grep "Aug" | sort
```

component

connector

Style 6: pipe-and-filter

- Key benefits
  - Modularity
  - Reuse
  - Anticipation of Change

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

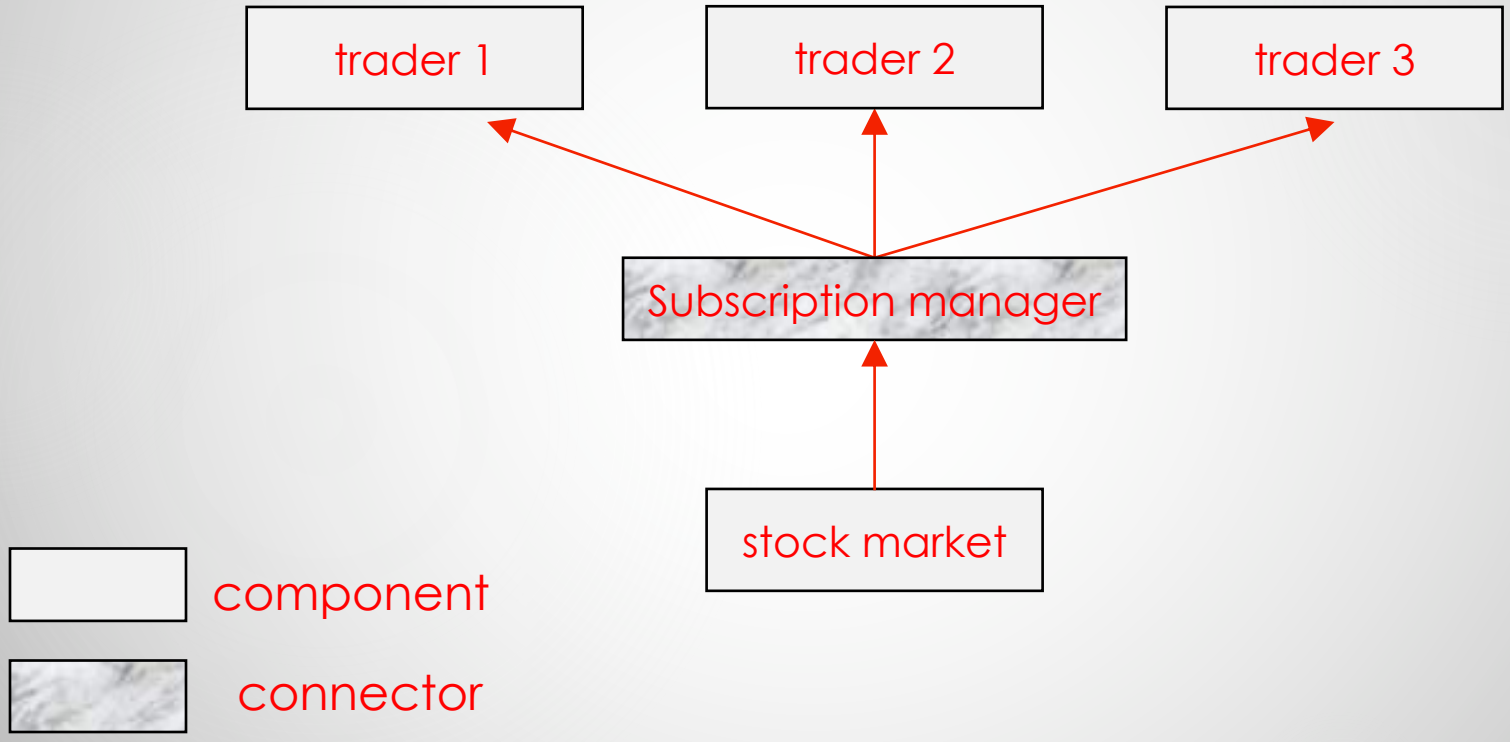
- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

Example 7: Stock Market Ticker

Goals:

- Filter through enormous amounts of info (stock prices) and only broadcast info that a subscriber is interested in

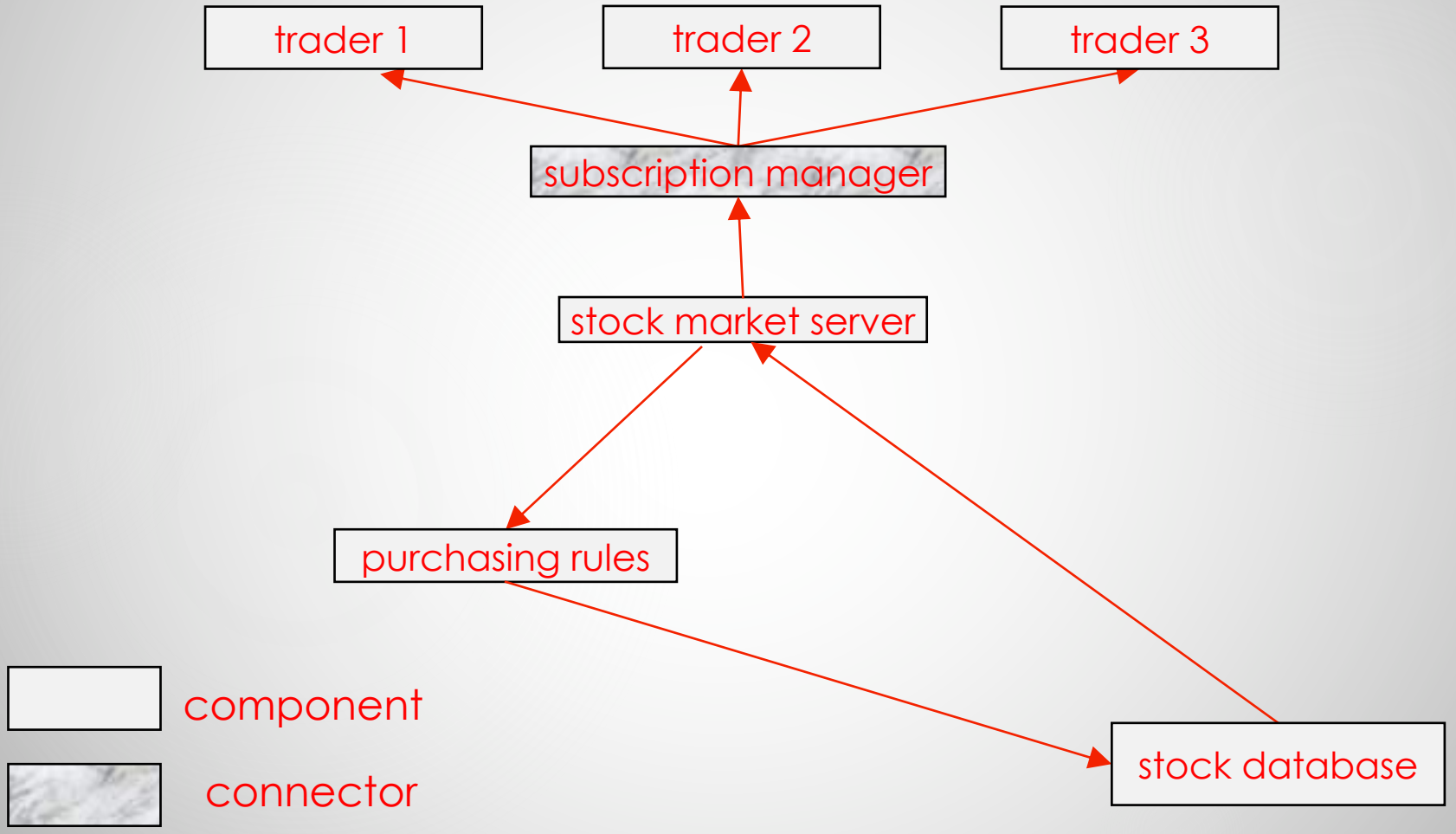
Style 7: publish-subscribe ("pub-sub")



Style 7: pub-sub

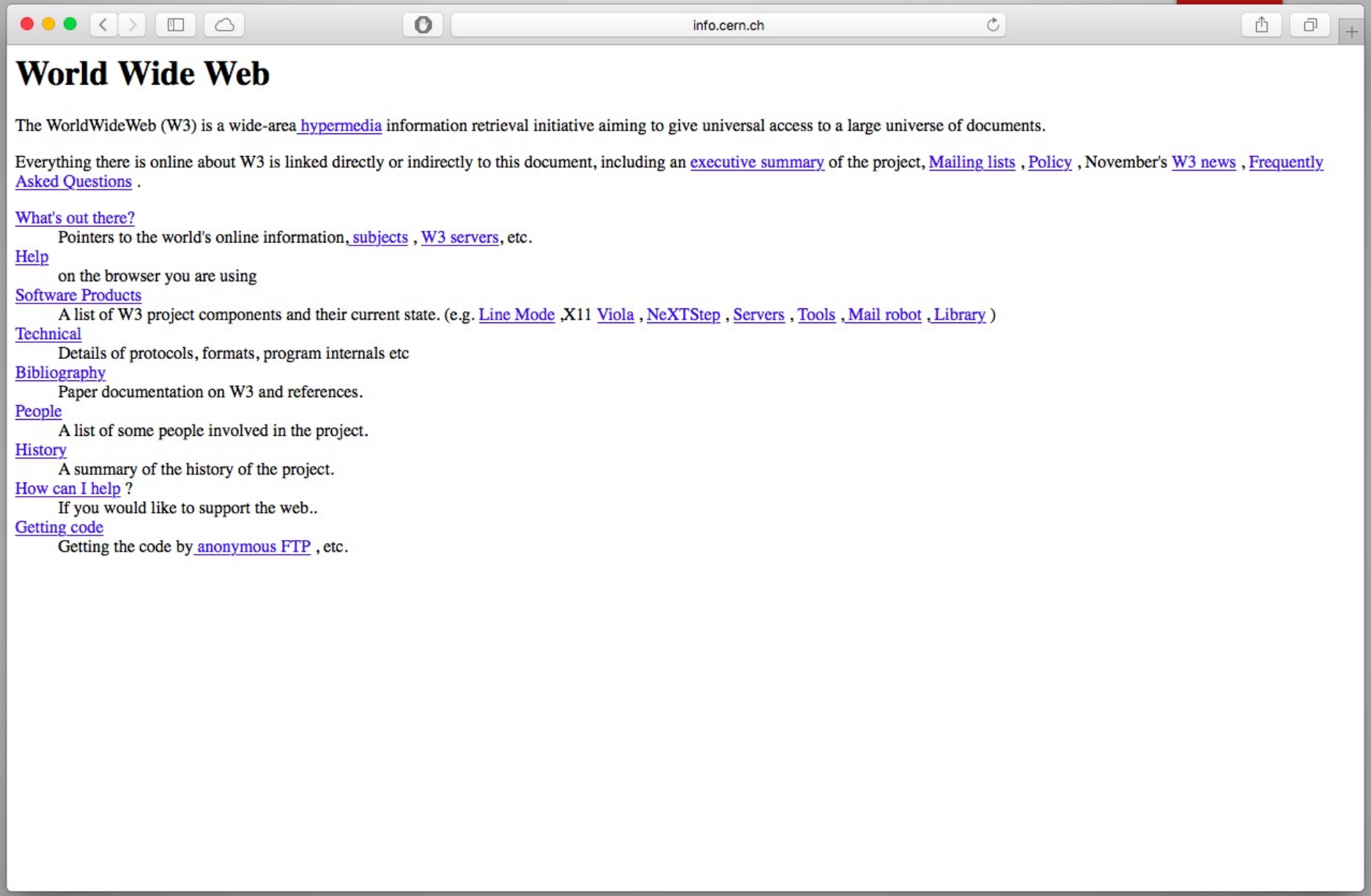
- Key benefits
  - Efficiency
  - Scalability

Mixing styles is often necessary



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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics


info.cern.ch
World Wide Web
The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.
Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .
What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.
Help
on the browser you are using
Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)
Technical
Details of protocols, formats, program internals etc
Bibliography
Paper documentation on W3 and references.
People
A list of some people involved in the project.
History
A summary of the history of the project.
How can I help ?
If you would like to support the web..
Getting code
Getting the code by anonymous FTP , etc.

Welcome to Amazon.com Books!

One million titles,
consistently low prices.

(If you explore just one thing, make it our personal notification service. We think it's very cool!)

Spotlight! -- August 16th

These are the books we love, offered at Amazon.com low prices. The spotlight moves **EVERY** day so please come often.

One Million Titles

Search Amazon.com's million title catalog by author, subject, title, keyword, and more... Or take a look at the books we recommend in over 20 categories... Check out our customer reviews and the award winners from the Hugo and Nebula to the Pulitzer and Nobel... and bestsellers are 30% off the publishers list...

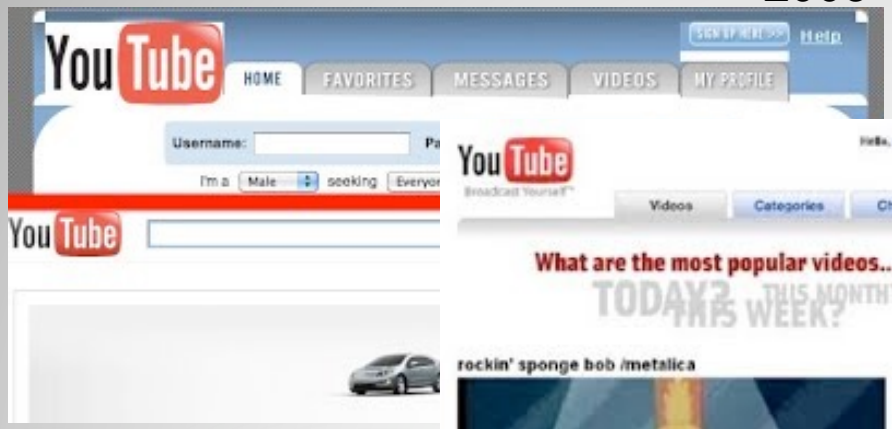
Eyes & Editors, a personal notification service

Like to know when that book you want comes out in paperback or when your favorite author releases a new title? Eyes, our tireless, automated search agent, will send you mail. Meanwhile, our human editors are busy previewing galleys and reading advance reviews. They can let you know when especially wonderful works are published in particular genres or subject areas. Come in, meet Eyes, and have it all explained.

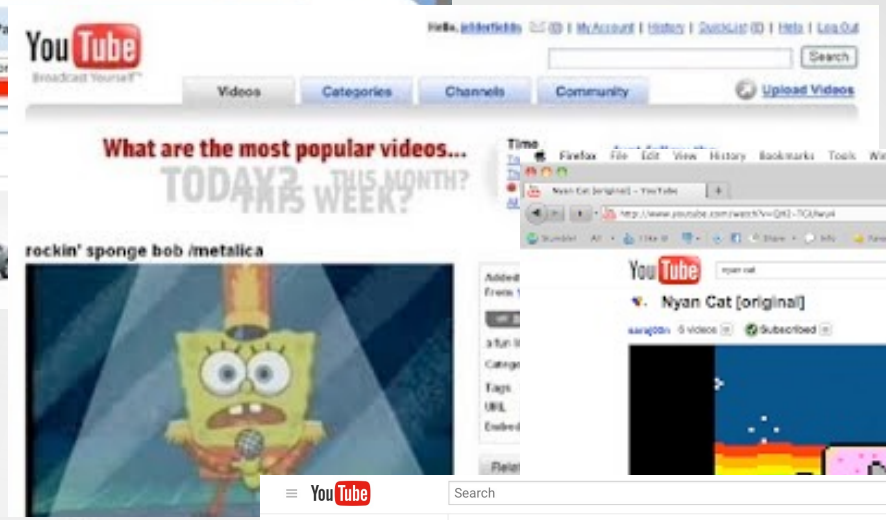
Your Account

Check the status of your orders or change the email address and password you have on file with us. Please note that you **do not** need an account to use the store. The first time you place an order, you will be given the opportunity to create an account.

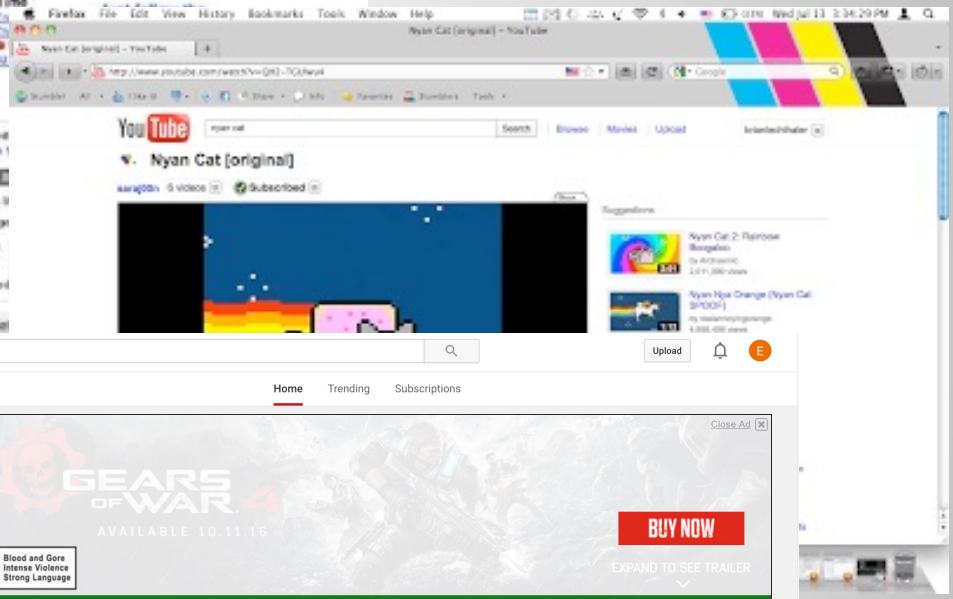
2005



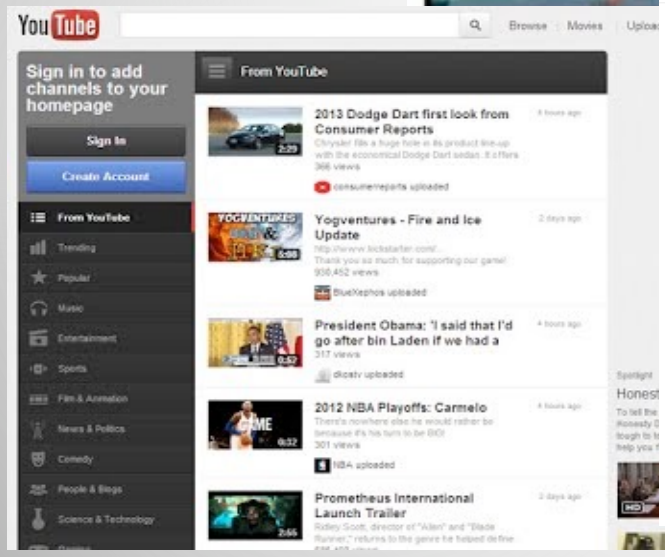
2007



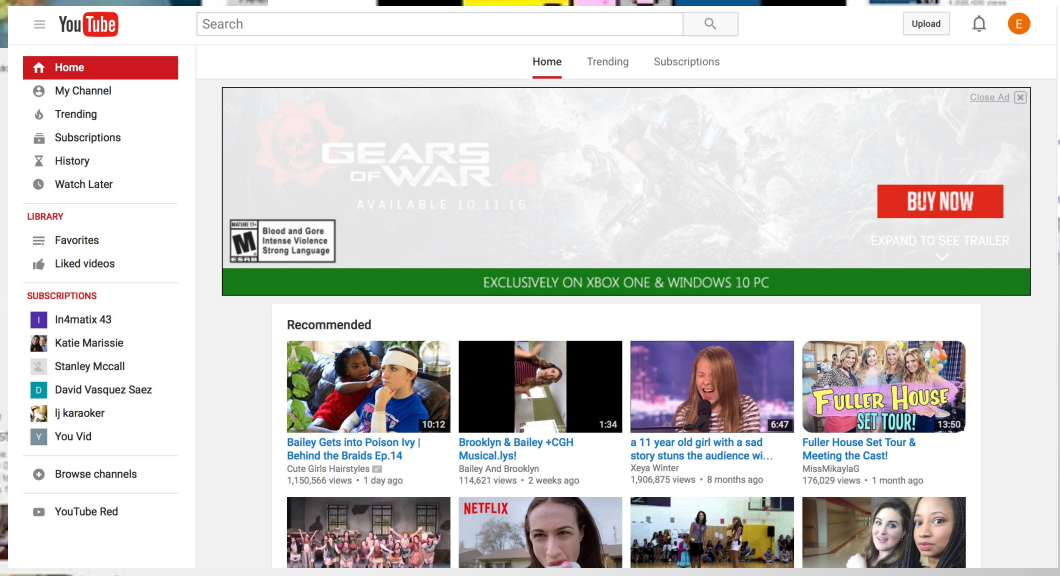
2011



2012



2016



Netflix

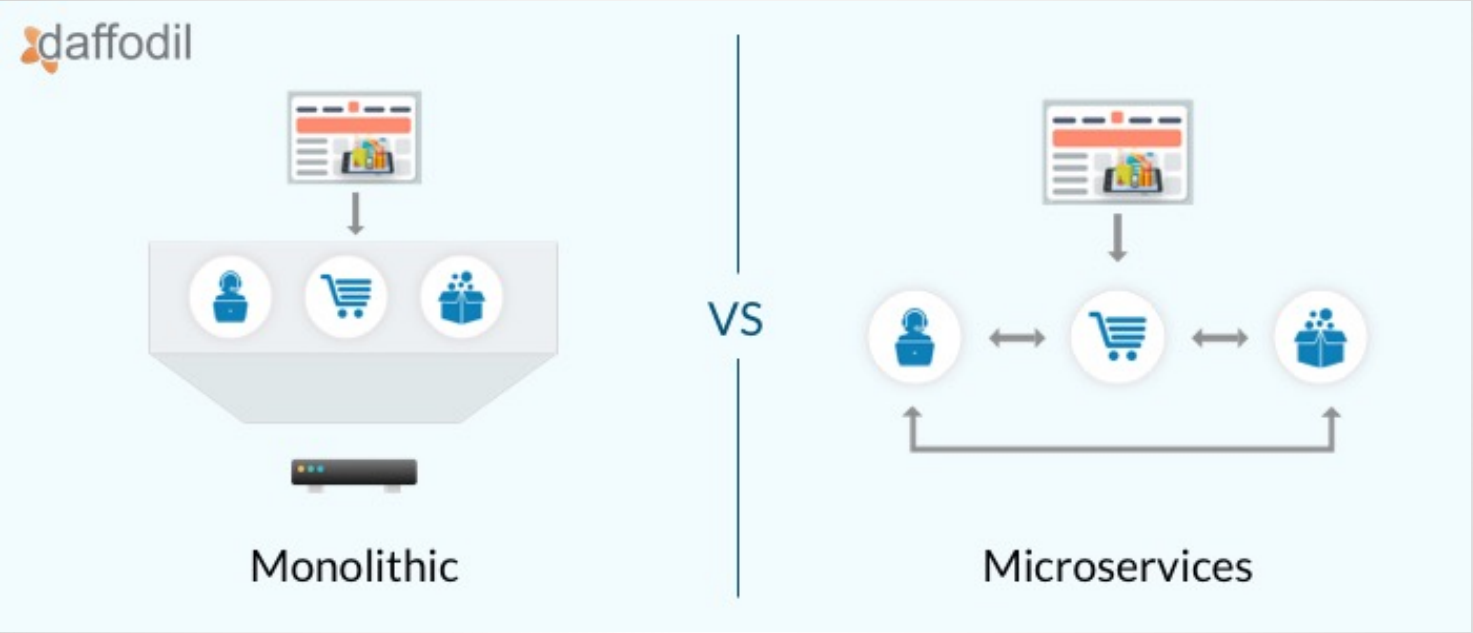
Source: https://media.netflix.com/en/company-blog/completing-the-netflix-cloud-migration

Evolution

- All successful software evolves!
- Essential part of software development
- Must be accommodated/planned as much as possible

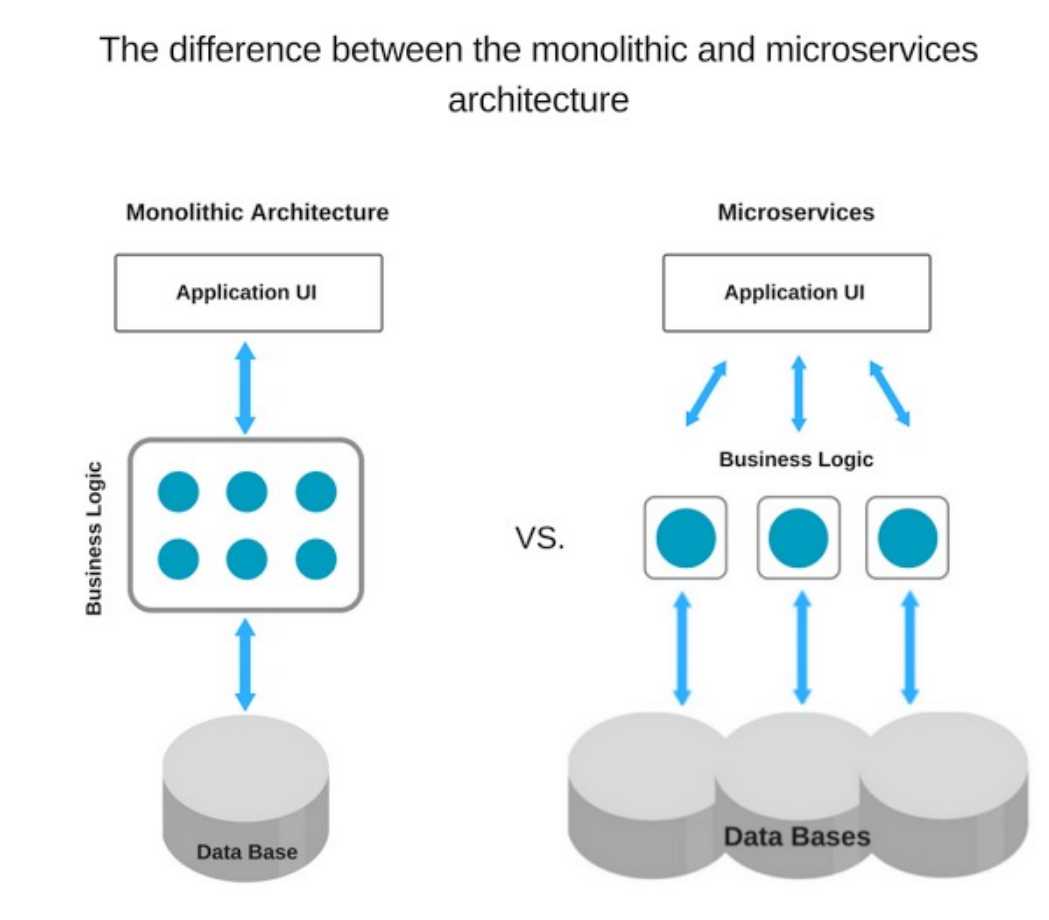
Architecture is a key tool in accommodating evolution

Trend: From Monolithic Towards Microservices



Source: https://insights.daffodilsw.com/blog/monolithic-vs-microservices-which-is-the-better-architecture-for-ecommerce-app-development

Trend: From Monolithic Towards Microservices



Source: https://www.altexsoft.com/blog/engineering/using-microservices-for-legacy-system-modernization/

Reminder: Recurring, fundamental principles

- Rigor and formality
- Separation of concerns
  - modularity
  - divide and conquer
  - abstraction
- Anticipation of change
- Generality
- Incrementality

How do these relate to software architecture?

Attendance Quiz: Architectural Styles Review

Summary

- An architectural style is
  - a named collection of architectural design decisions that result in beneficial qualities in each resulting system
- Styles: model-view-controller, client-server, layered, peer-to-peer, pipe and filter, pub-sub
  - (These are not all the styles that exist)
- All successful software evolves
  - We must plan for this
  - Architecture helps us do so

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

- Architectural styles
  - Model-view-controller
  - Client-server
  - Layered
  - Peer-to-peer
  - Pipe-and-filter
  - Publish-subscribe
- Evolution
- Quiz 3 Topics

Quiz 3 - Topics

- Use Cases
  - Actors
  - Flows (basic/alternative/exception)
  - Use case descriptions
  - Use case diagrams
  - Purposes/uses of use cases
- Understand what software architecture is
- Elements of software architecture
- Architectural erosion (prescriptive vs. descriptive architectures)
- Know and understand the 6 architectural styles that we discussed

Next time

- Designs, Models, and Notations
- Readings (linked on Canvas)