EBU6304 – Software Engineering

Software Architecture

- Topics
  - Software architecture
  - Agility and architecture
  - Architectural patterns
    - Web-based architecture
    - Distributed systems architecture
    - RESTful architecture
    - Mobile applications architecture

Software architecture

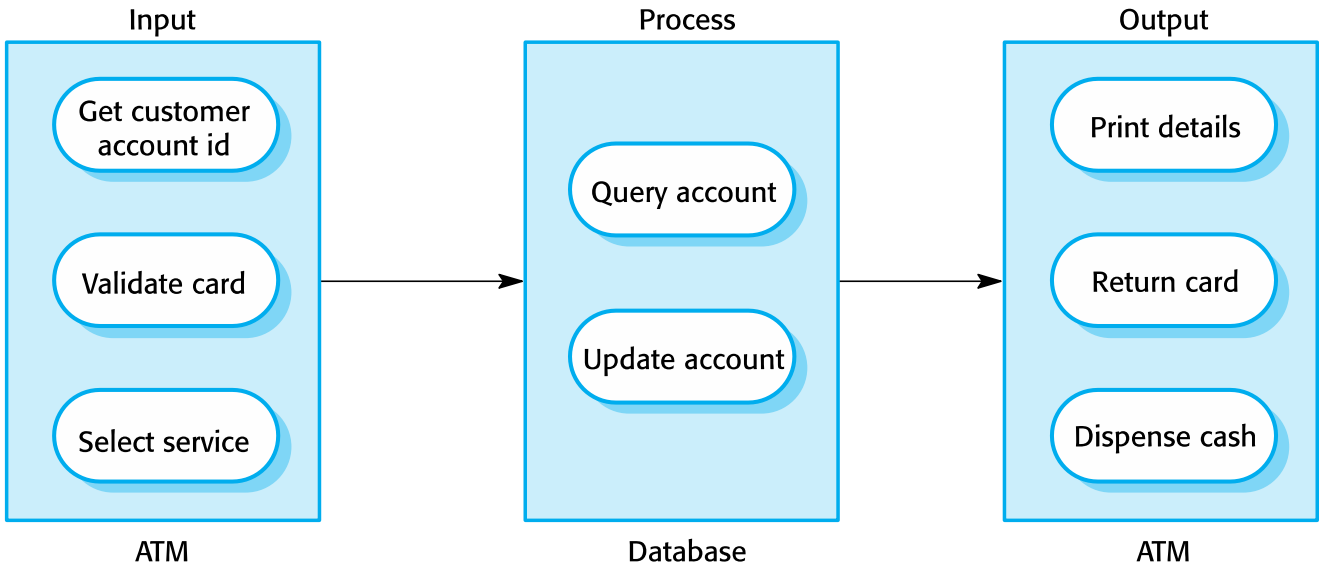
- Software architecture refers to the set of principal design decisions, the blueprint of a software system

Software architecture

- Planning the architecture of a software system means understanding how the system should be organized and how to design its overall structure
- Requirements analysis and software architecture are inevitably connected
  - Non-functional requirements are a result of the architectural choices of the system

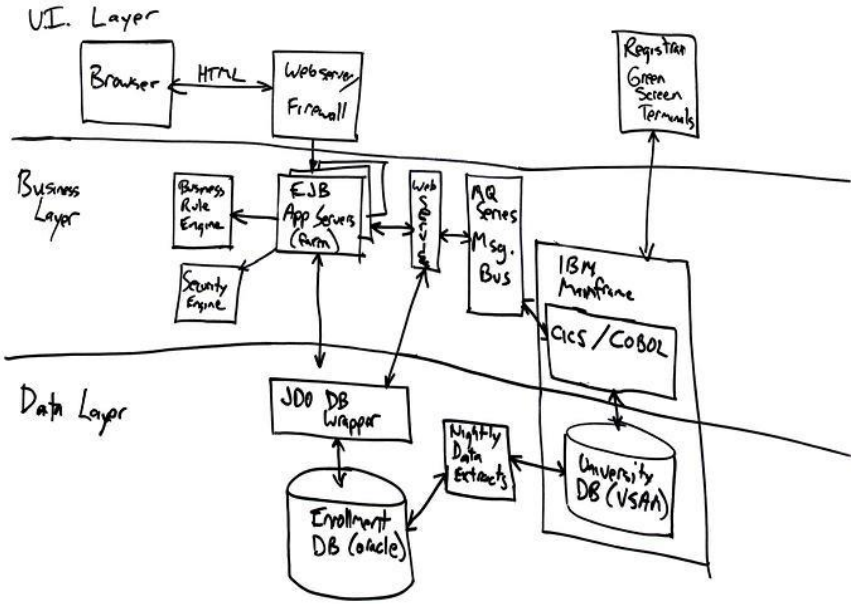
Example: SA of an ATM system

- Separation of Components (computation) and Connectors (communication)
  - An example of the application of the fundamental SE design principle "separation of concerns"



Architectural models

- The architecture is often represented using a simple box and arrow model, or a UML diagram (e.g., component diagram)



http://agilemodeling.com/essays/initialArchitectureModeling.htm

Architectural models

- There exist also specialised languages/models for architecture specification however they are beyond the scope of this module
  - Allow to perform more advanced analyses
  - e.g., Darwin architectural description language

https://en.wikipedia.org/wiki/Darwin_(ADL)

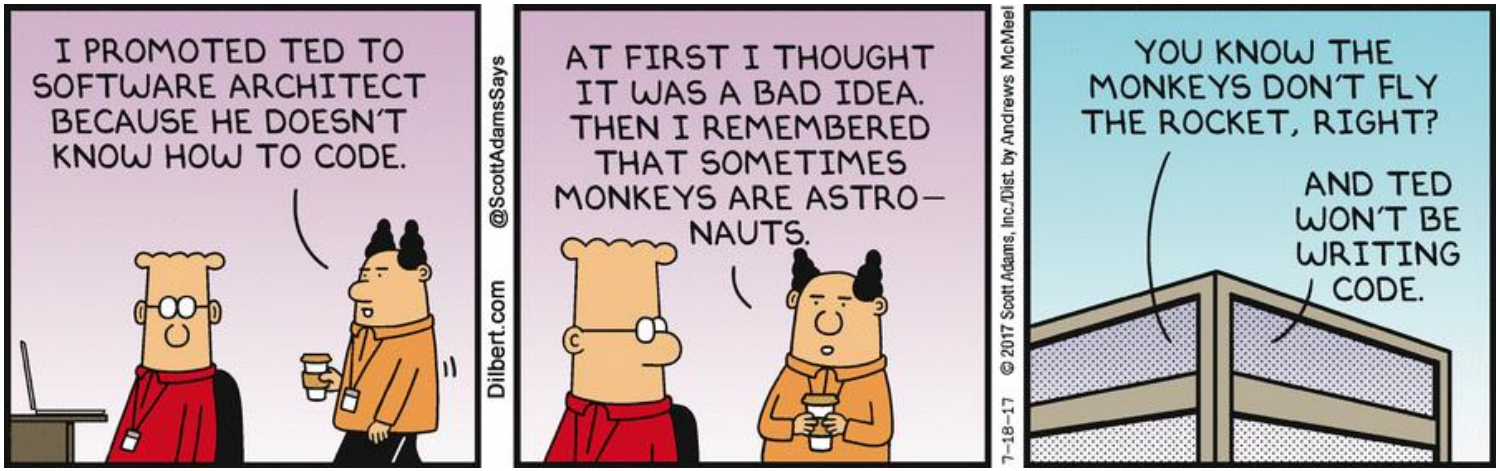
```
component DataStore{
    provide landerValues;
}
component Calculation{
    require landerValues;
    provide calculationService;
}
component UserInterface{
    require calculationService;
    require landerValues;
}
component LunarLander{
inst
    U: UserInterface;
    C: Calculation;
    D: DataStore;
bind
    C.landerValues -- D.landerValues;
    U.landerValues -- D.landerValues;
    U.calculationService --
C.calculationService;
}
```

Advantages of focusing on architecture

- System analysis
  - The analysis of whether the system can meet its non-functional requirements is possible
- Large-scale reuse
  - The architecture may be reusable across a range of systems (reuse is sometimes an economic necessity)
  - Product-line architectures may be developed
- Stakeholder communication
  - Architecture may be used as a focus of discussion by system stakeholders

Agility and architecture

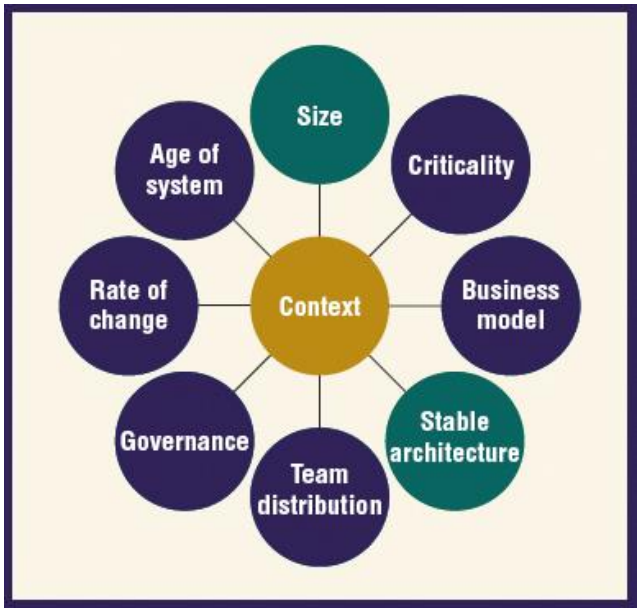
- Agile development advocates early delivery and rapid response to change, how can we afford to spend time without coding and designing the architecture instead?



Agility and architecture

- It is generally accepted that an early stage of agile processes is to design the overall architecture of the system
- This is because the system architecture is usually expensive to change because it affects so many important components in the system
- The architectural design does not need to focus on all design elements of the software but only the most important ones

Agility and architecture



"Some factors making up a project's context. Like other software design and implementation activities, the project's context, including the customer, needs to drive the project's architectural activities."

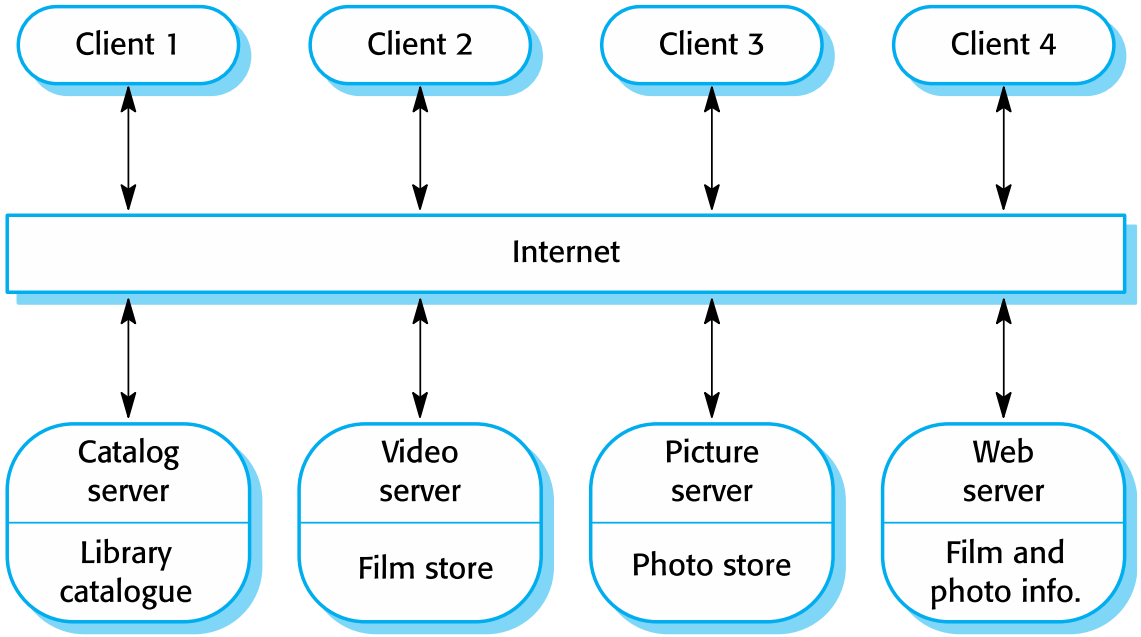
From "Agility and Architecture: Can They Coexist?"
https://www.computer.org/csdl/magazine/so/2010/02/mso2010020016/13rRUy0qnJM

Architectural patterns (or styles)

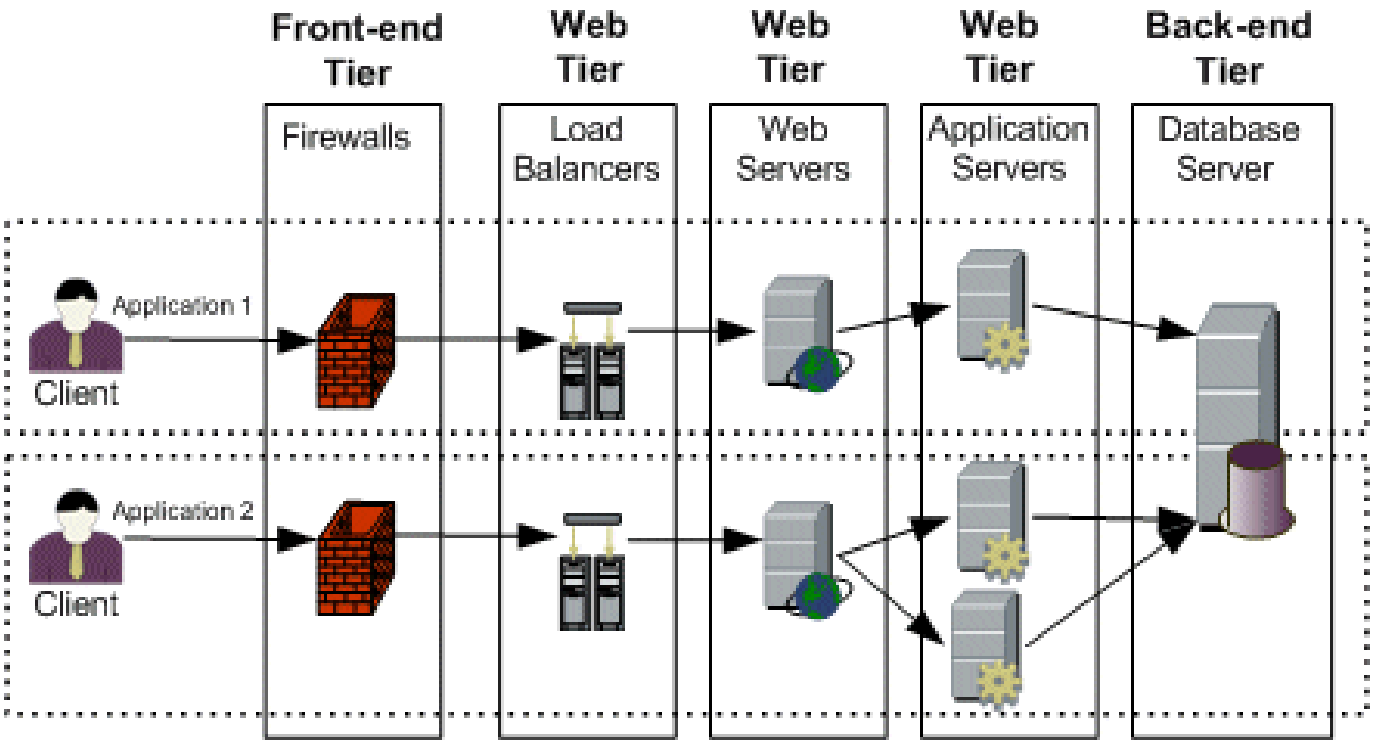
- Patterns are a means of representing, sharing and reusing knowledge
- An architectural pattern is a stylized description of good design practice, which has been tried and tested in different environments
- Patterns should include information about when they are and when they are not useful
- Patterns may be represented using tabular and graphical descriptions

Web-based architectures

- Web-based applications are usually structured according to the client-server architecture, potentially organised into multiple tiers

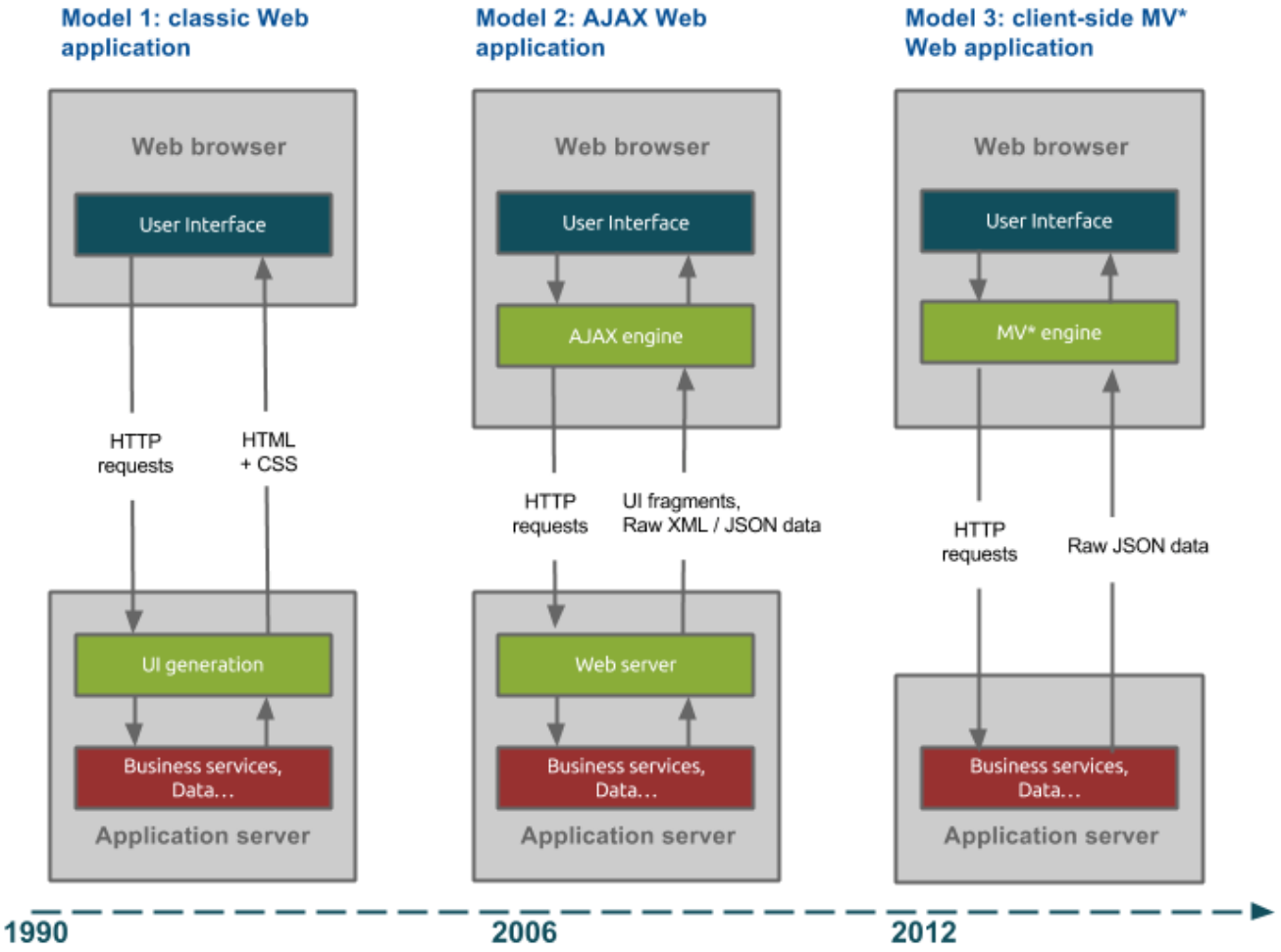


Web-based architectures



https://bsm.asggroup.com.au/topaz/amdocs/eng/doc_lib/Content/EUM/AdminGuide/back_end_web_tiers.htm

Web-based architectures



https://blog.octo.com/en/new-web-application-architectures-and-impacts-for-enterprises-1/

Distributed systems architecture

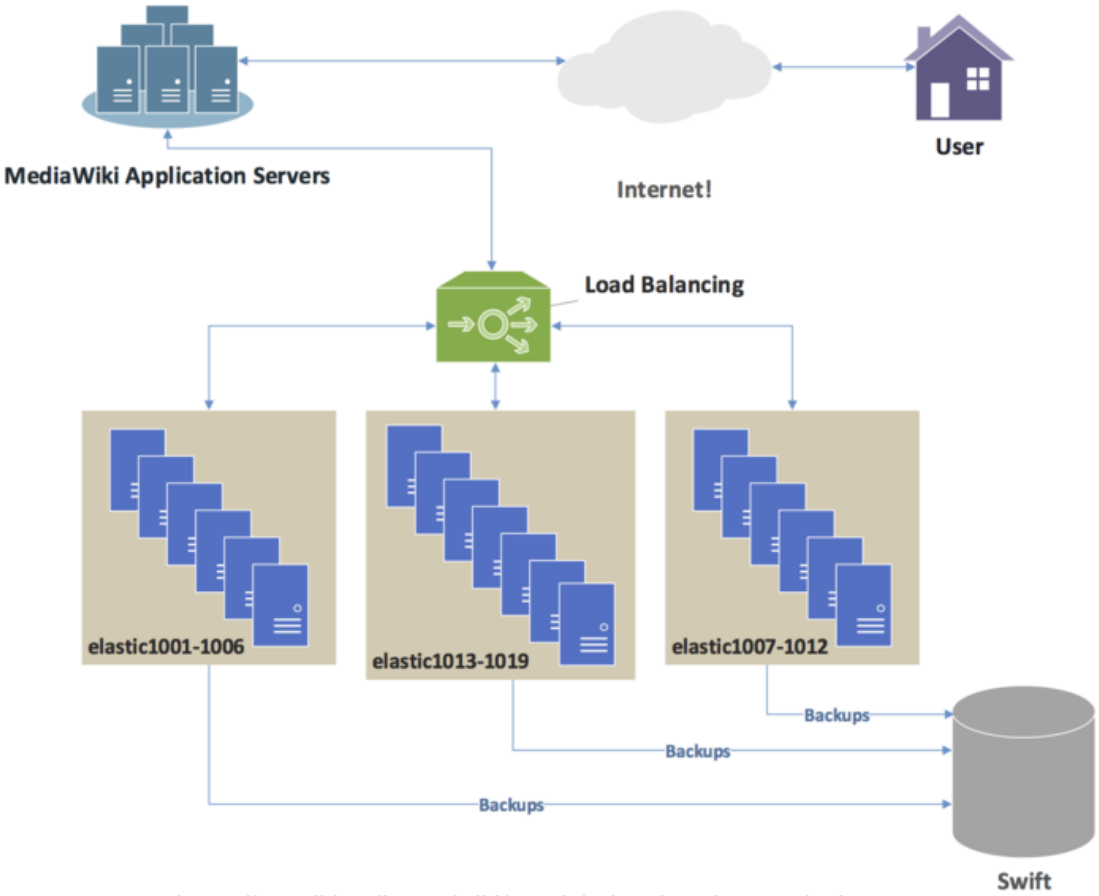
- Client-server architecture suffers from the presence of a single point of failure and bottleneck
- Distributed systems are designed to overcome this by relying on a variety of techniques, including:
  - Replication and clusters
  - Load balancing
  - Caching
  - Serverless/cloud computing
  - Hadoop and Map Reduce

Cloud computing

- Cloud computing refers to the delivery of on-demand computing services (software and hardware) typically over the Internet

http://gnoted.com/what-is-cloud-computing-simple-terms/

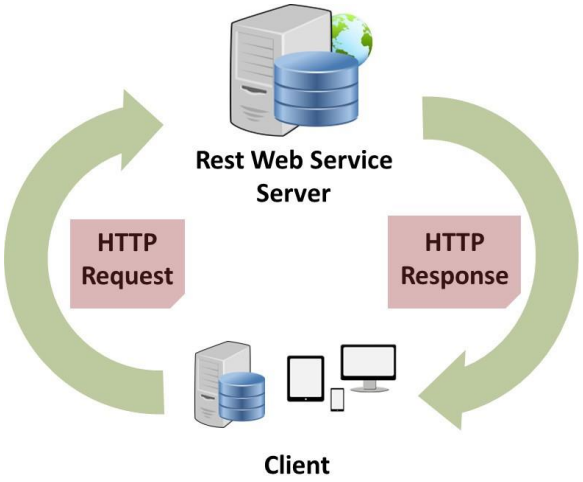
Load balancing



https://en.wikipedia.org/wiki/Load_balancing_(computing)

RESTful architecture

- Representational state transfer (REST) is a software architectural style that defines how to structure Web services, so that textual representations of Web resources can be accessed and manipulated using a uniform and predefined set of stateless operations

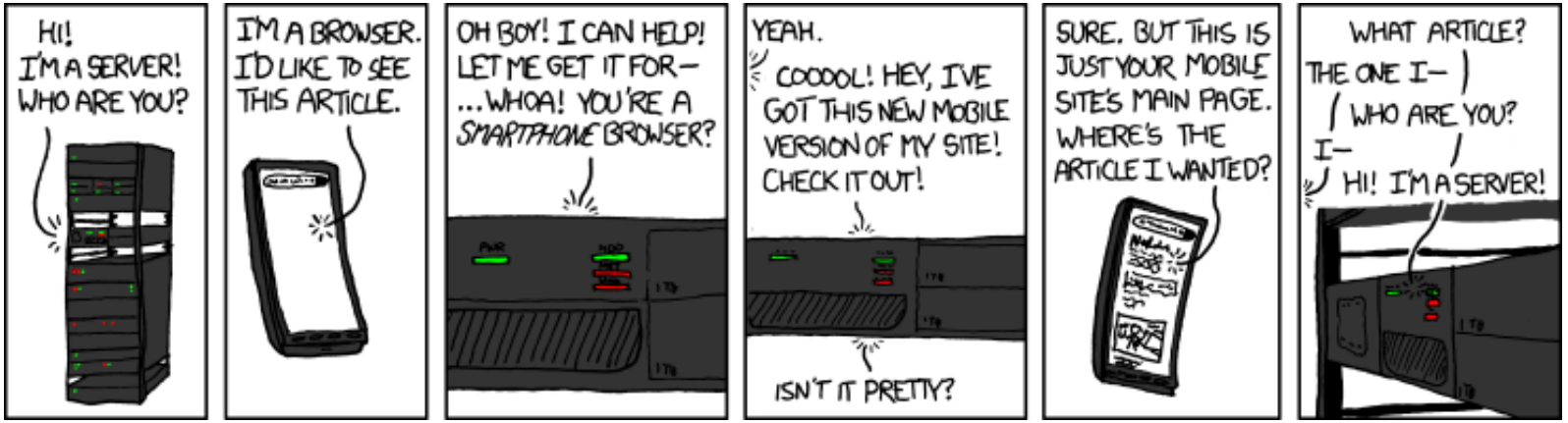


https://medium.com/@ahmetozlu93/mastering-rest-architecture-rest-architecture-details-e47ec659f6bc

RESTful architecture

- REST is a combination of 6 architectural constraints

(1) Client-server
(2) Cacheability
(3) Uniform interface
(4) Statelessness
(5) Layered system
(6) Code-On-Demand

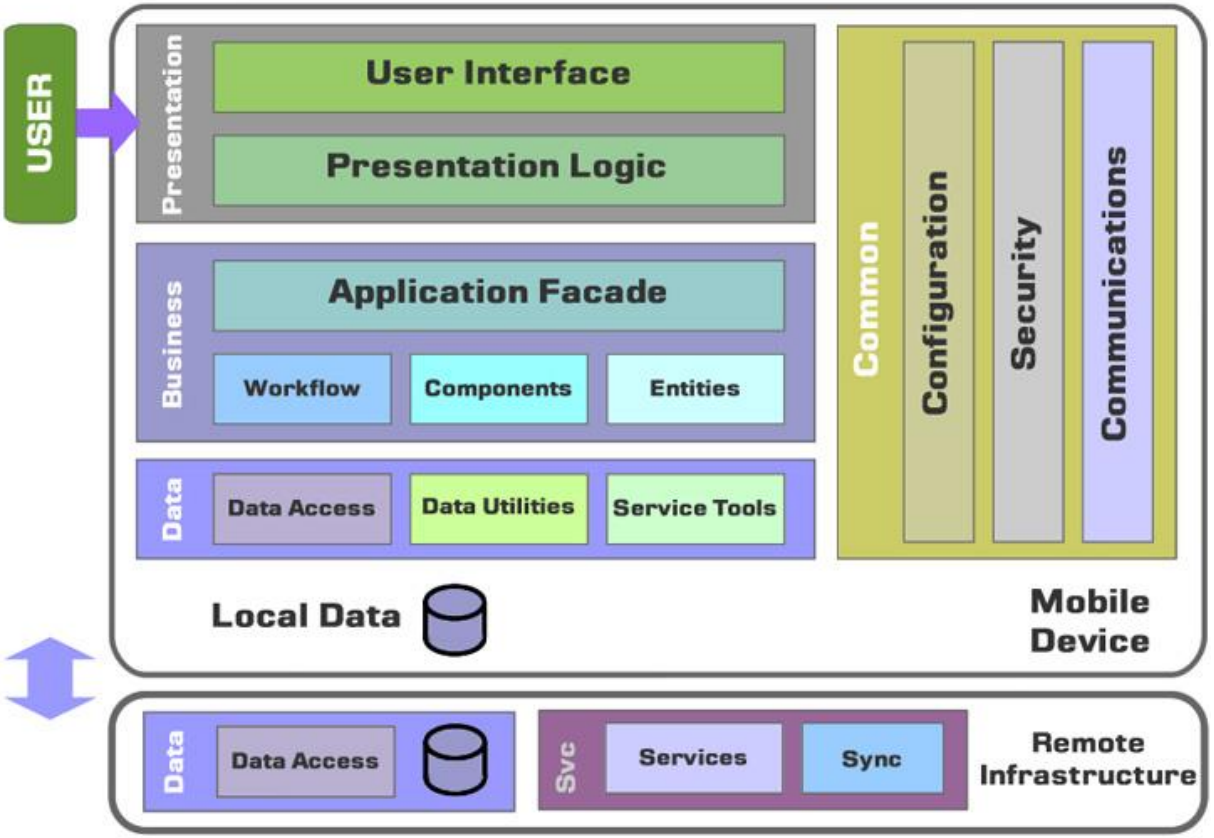


https://www.explainxkcd.com/wiki/index.php/869:_Server_Attention_Span

Mobile applications architecture

- Applications developed for mobiles are usually structured using a layered architecture
  - *Presentation layer*
    - User Interface and UI process components
  - *Business layer*
    - Main functionalities, potentially deployed on a remote backend to reduce the load on the mobile
  - *Data layer*
    - Data helpers and utilities, data access components, and service agents

Mobile applications architecture



https://leadingmobilenews.com/everything-you-need-to-know-about-mobile-app-architecture/

Summary

- A software architecture is a description of how a software system is organized
- The design decisions that concern the architecture are the most important ones, the ones that are harder to change at a later stage
- Architectural patterns are a means of reusing knowledge about generic system architectures, making systems safer and faster to implement
- It is generally accepted that an early stage of agile processes is to design the overall architecture of the system

References

- Chapter 6 – “Software Engineering” textbook by Ian Sommerville
- Agility and Architecture: Can They Coexist? https://www.computer.org/csdl/magazine/so/2010/02/mso2010020016/13rRUy0qnJM
- REST API tutorial https://restfulapi.net/