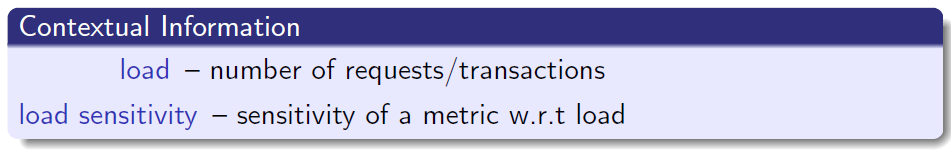
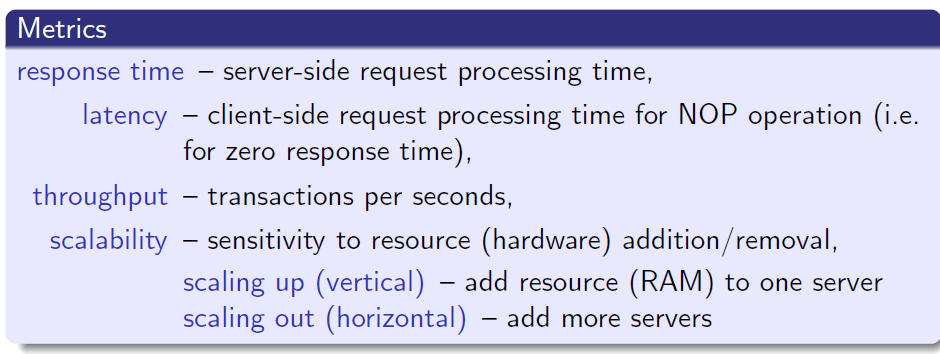
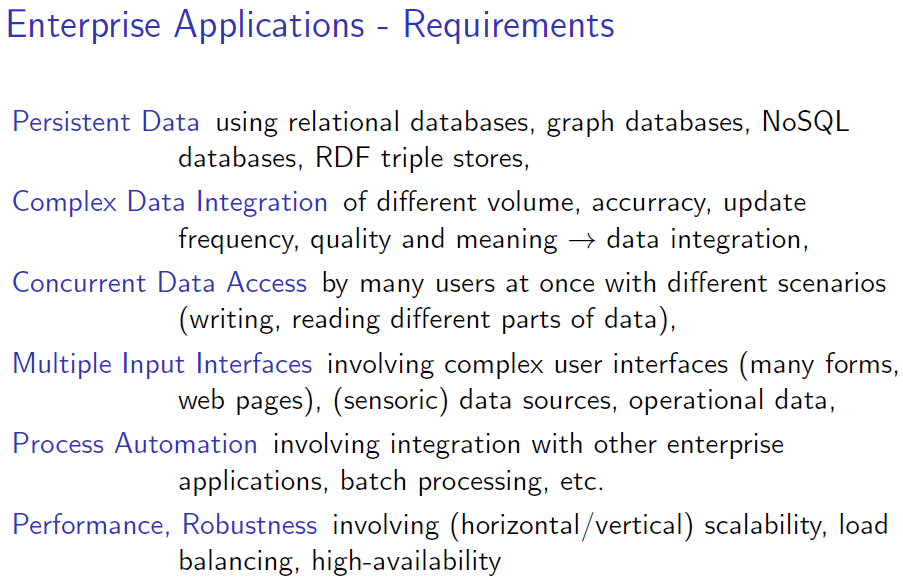
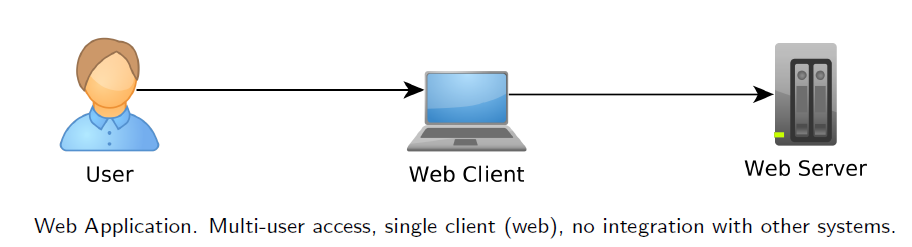
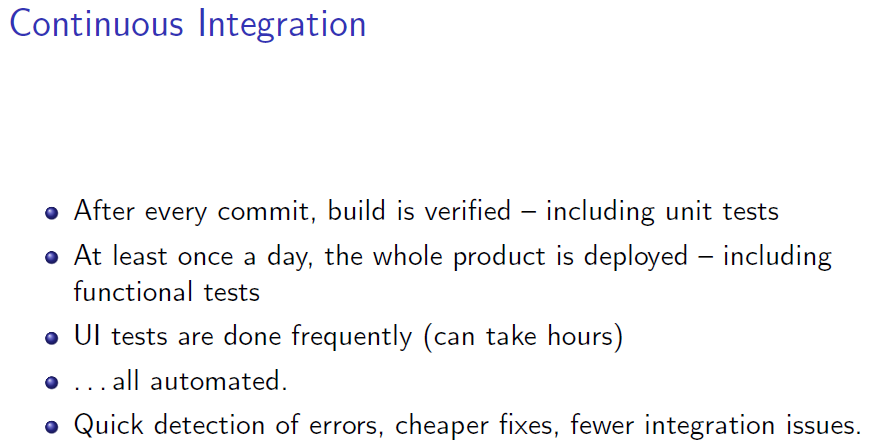
EAR

Architektura enterprise aplikací, vrstevnatá architektura, hexogonální architektura, inversion of control, dependence injection, beans a jejich životní cyklus, bezpečnost eterprise aplikací. (EAR)

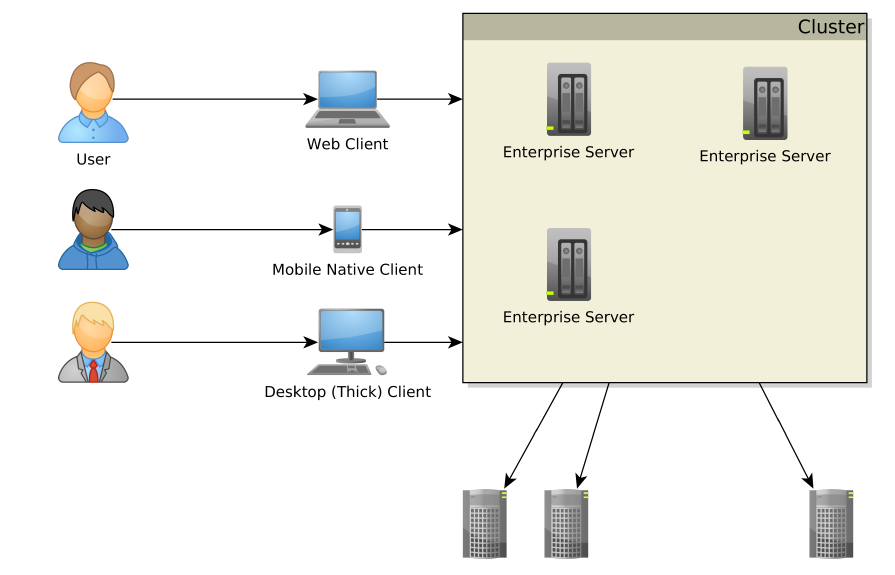
# Základní informace



Simple webserver



Enterpreise architecture

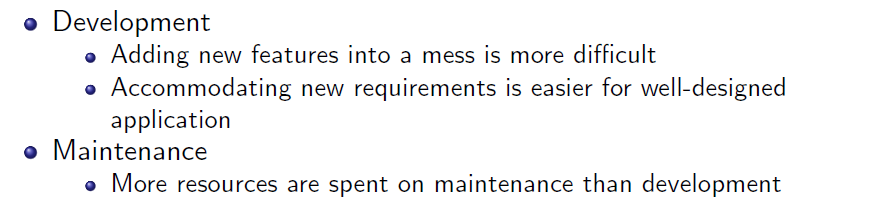


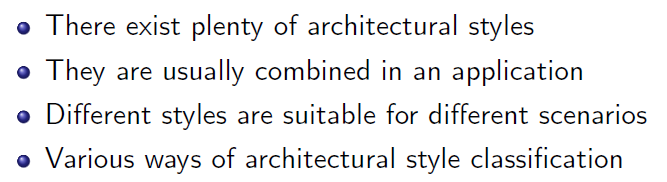
# Architektura

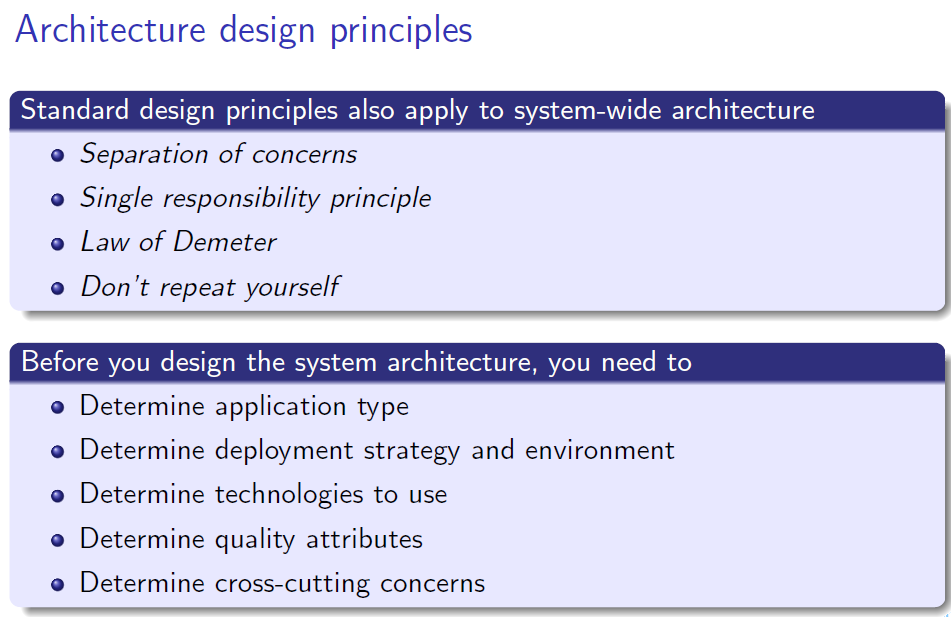
The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those

elements, and the relationships among them. Architecture is concerned with the public side of interfaces; private details of elements|details having to do solely with internal

implementation are not architectural.



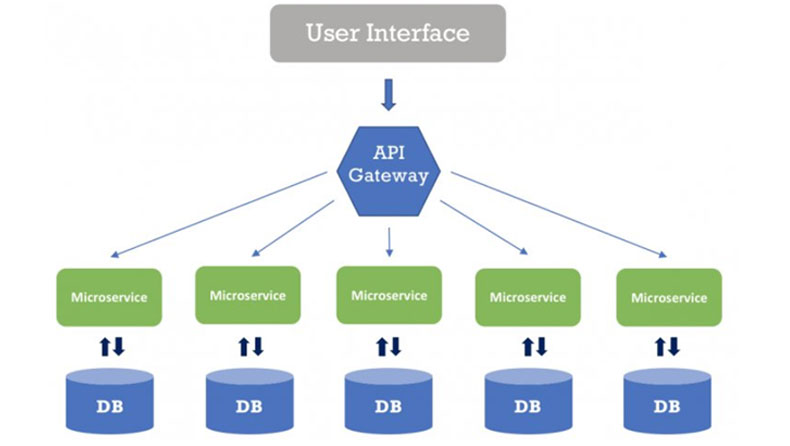




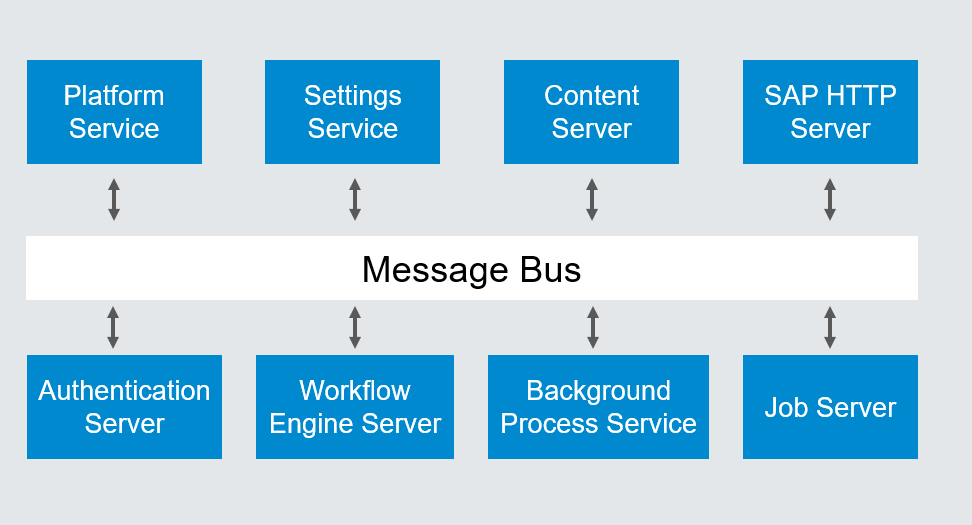
## Architecture – Communication

### **Service-Oriented Architecture (SOA)**

* Neexistuje centrální bod komunikace - servicy spolu komunikují přímo
  + Při výpadku providera - chyba
  + Není možné realizovat další logiku (např. překlad komunikace (SOAP - REST / REST - SOAP))
* Rozlišuje v komunikaci providera a konzumenta
* Komunikují pomocí stejných technologií (schéma)
* **Microservices**
  + SOA dohnaná do největšího levelu granulity
  + Nemá přesnou definici



### **Message bus**

* “SOA s centrálním komunikačním busem”
* Má centrální bod komunikace – větší flexibilita
* Při výpadku umožňuje bufferovat zprávy
* **Enterprise Service Bus – ESB**
  + Umožňuje překládat komunikaci různých servis
  + Logika na middlewaru (např. propojení REST-SOAP a vice versa komunikace, routing, decoupling...)

## Architecture – Deployment

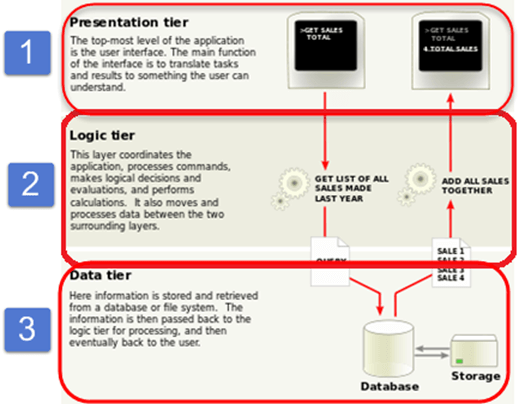
### **Client/server**

* Jasně daný klient, server
  + Server je jeden fyzický stroj (layered architecture)
* Possible single point of failure

### **N(3)-Tier**

* Nezávislé vrstvy poskytující funkcionalitu
* Jednoduché škálování
* Více fyzických strojů

The most common form of n-tier is the 3-tier Application, and it is classified into three categories.

* User interface programming in the user's computer
* Business logic in a more centralized computer, and
* Required data in a computer that manages a database.

### **Peer-to-Peer**

* Decentralizovaná architektura
* Klient je serverem i klientem
* Dělení
  + Unstructrured - bez centrální nody, discover peerů je časově náročné
  + Structured - daná topologie lepší získávání peerů
  + Hybrid

## Architecture – Domain

### **Domain Driven Design**

* Struktura a jazyk softwarového kódu odpovídá business doméně (názvy pojmů => třídy, metody, proměnné odpovídají business)
  + Customer
    - AcceptOffer()
    - Withdraw()

### **Model Driven Engineering**

* Větší zaměření na kód
* Reflektuje business doménu, ale zaměřuje se na větší smysl kódu

## Architecture – Structure

### **Object-Oriented**

* Objekt je model, z reálného nebo virtuálního světa. Má svoje vlastnosti, metody a události.
* Enkapsulace implementační detailů
  + Public / Private identifikátory přístupu

### **Component-Based**

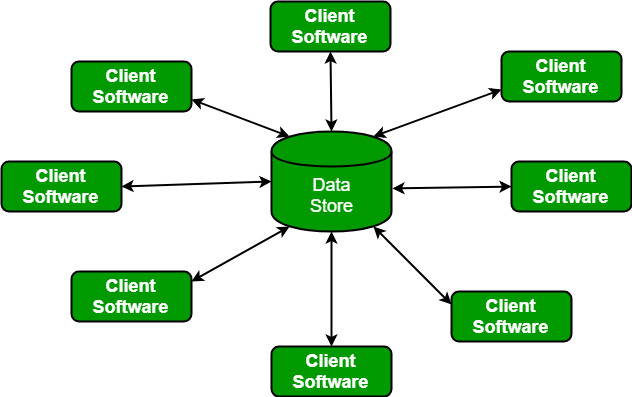
* Vyšší úroveň než objekty
* Poskytují interfaces
* Mohou být spravovány architecture providerem (Spring)
* Komponenty mohou být distribuované
* Separation of Concerns, encapsulation

### **Layered Architecture (N-tier)**

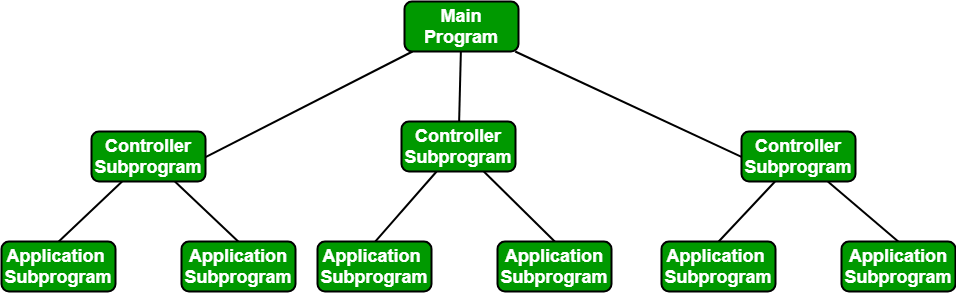
* Vrstva = related funkcionality
* Monolit = jeden stroj
* Komunikace pouze dolů nebo stejnej level
* Cross cutting concern

### **Data centric**

* Všechny aplikace v organizaci používají společný data model



### **Call and return**

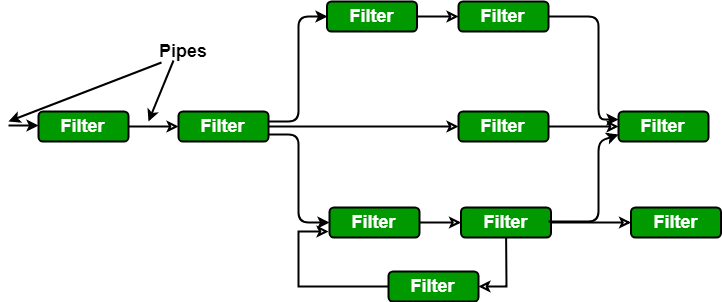


### **Implicit invocation**

* Postavené kolem
  + Events
  + Callbacks
* Komponenta může vyvolat více eventů
  + Event announcement implictly causes invocation of procedures in other module
* Eg. Observer pattern

### **Pipe & Filter**

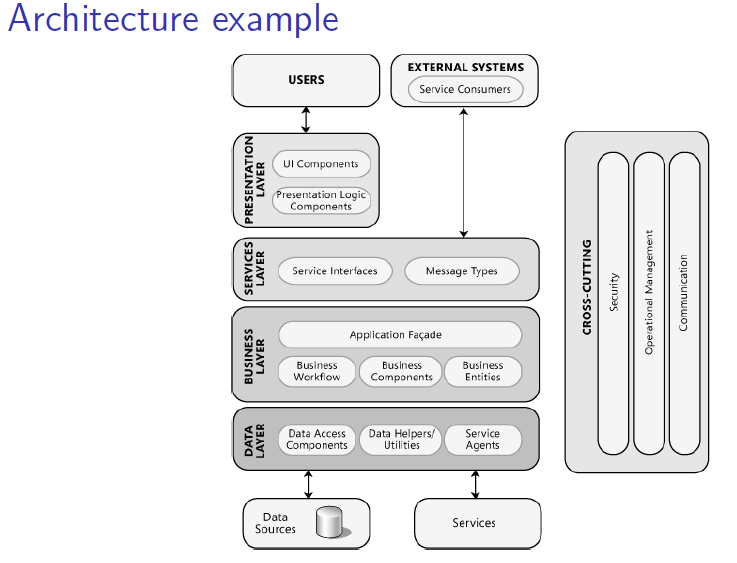
* Populární architektura pro Compiler, UNIX Shell (cat, grap a sort kombo) a např Java Stream



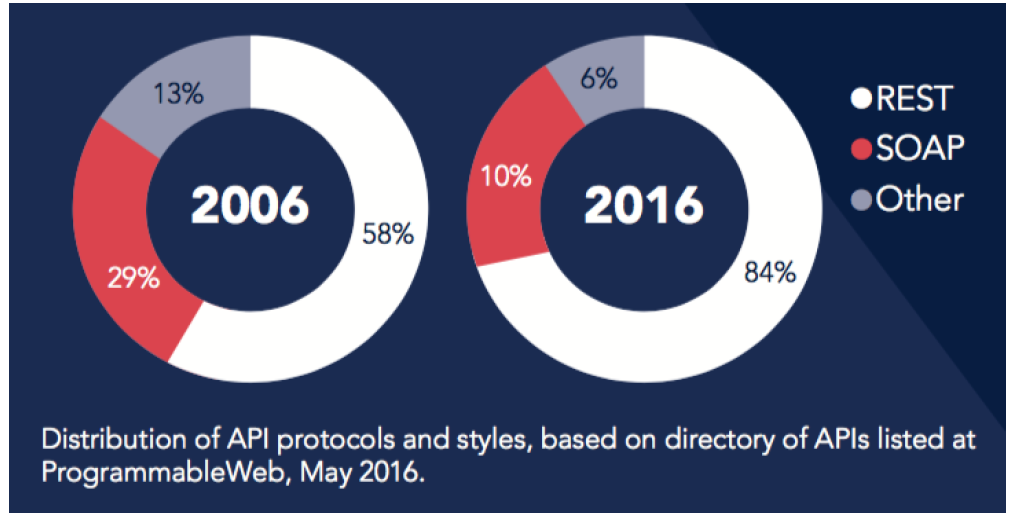
### **Inversion of controll**

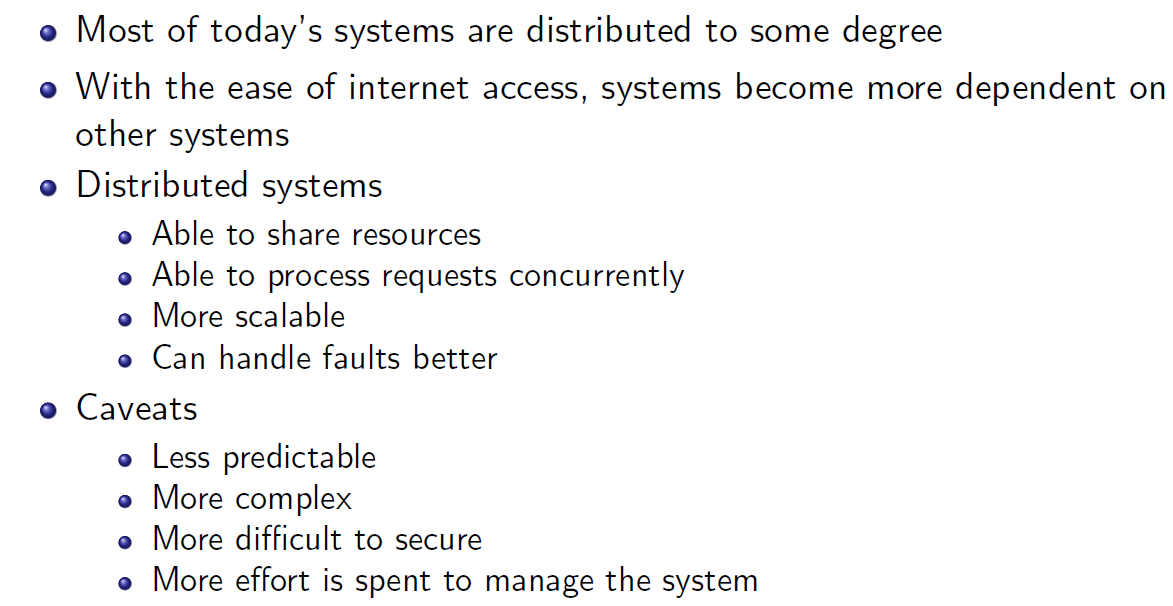
****

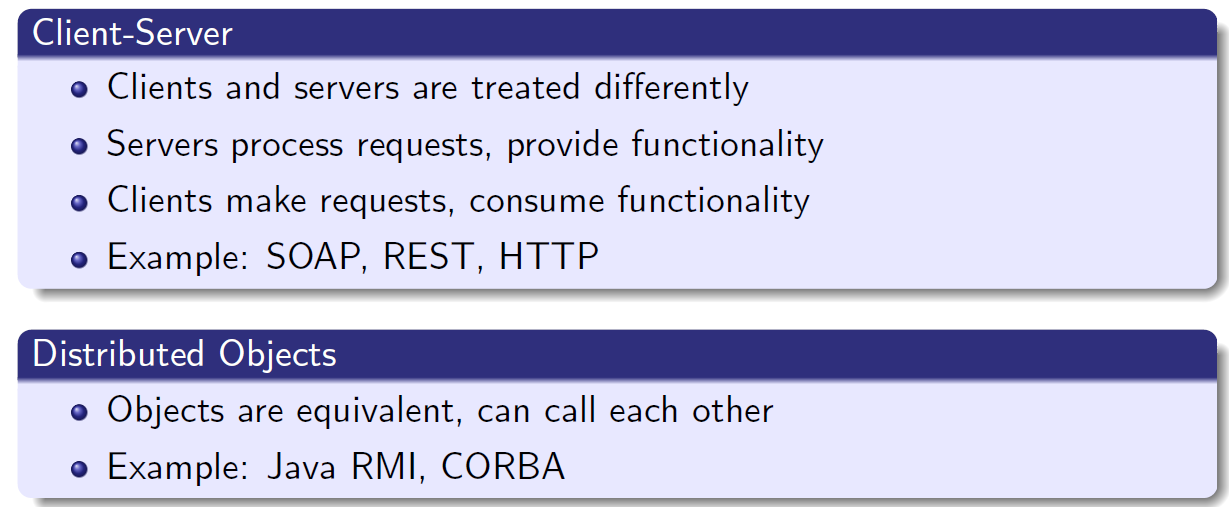




# REST a další rozhraní

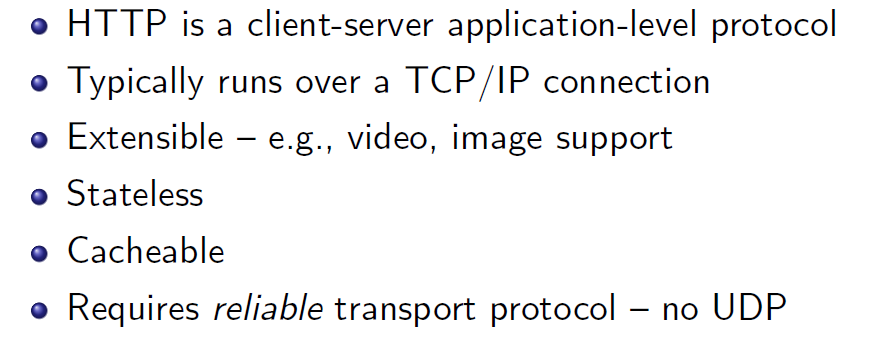
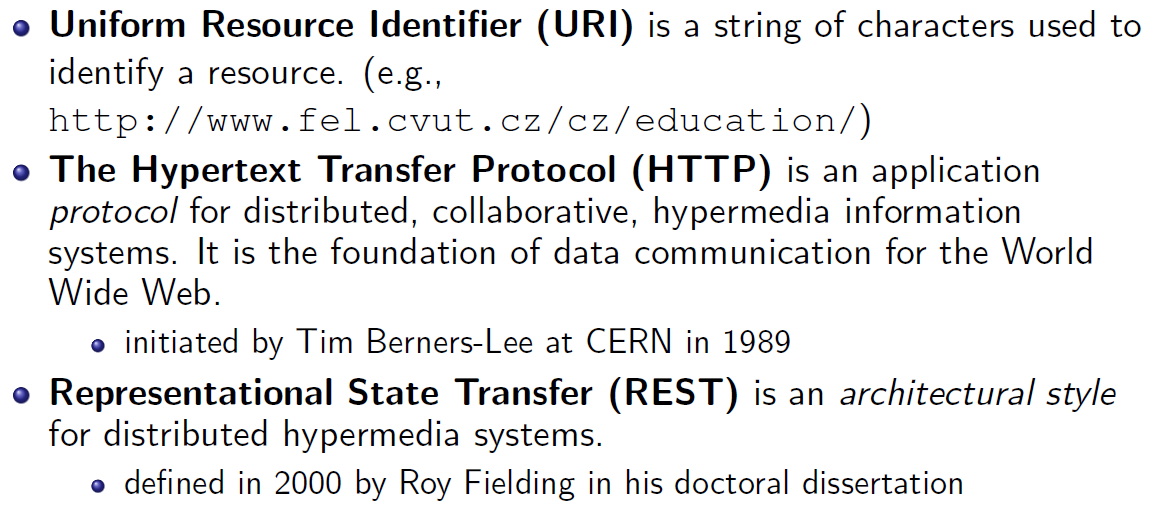
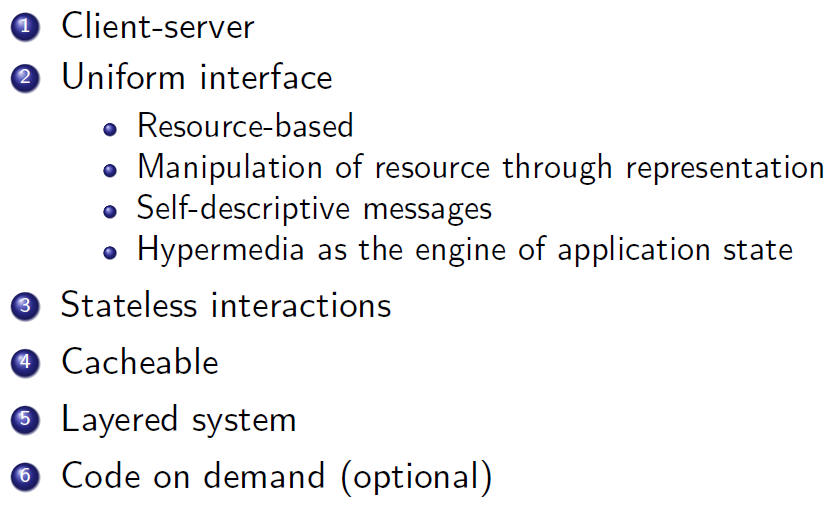
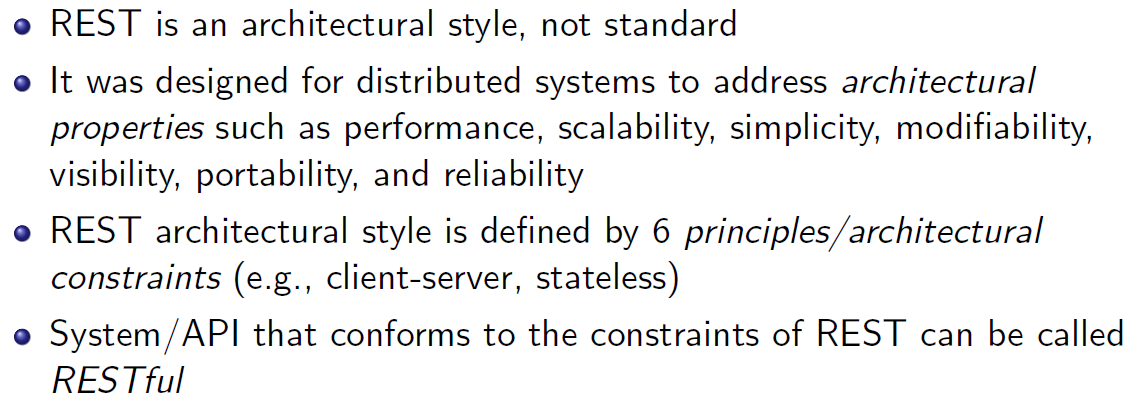


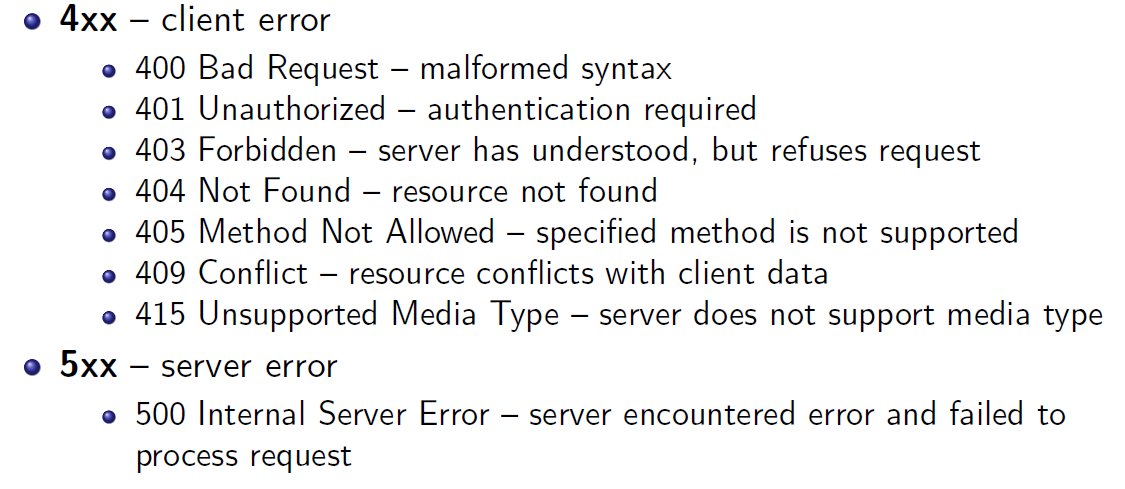
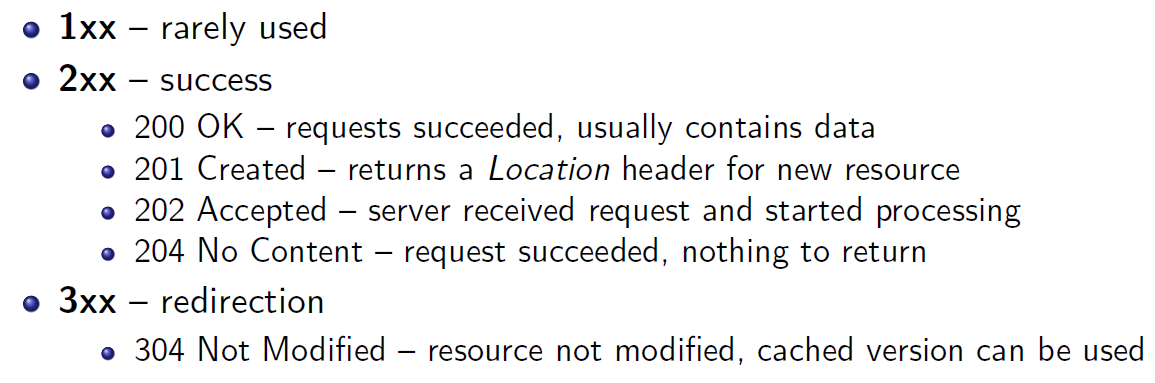
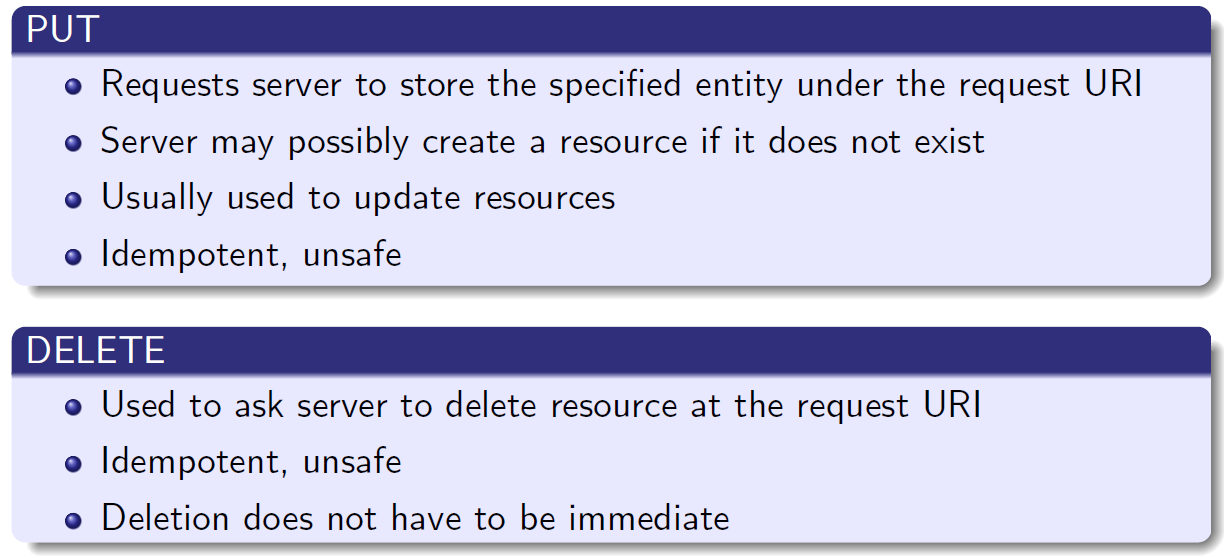
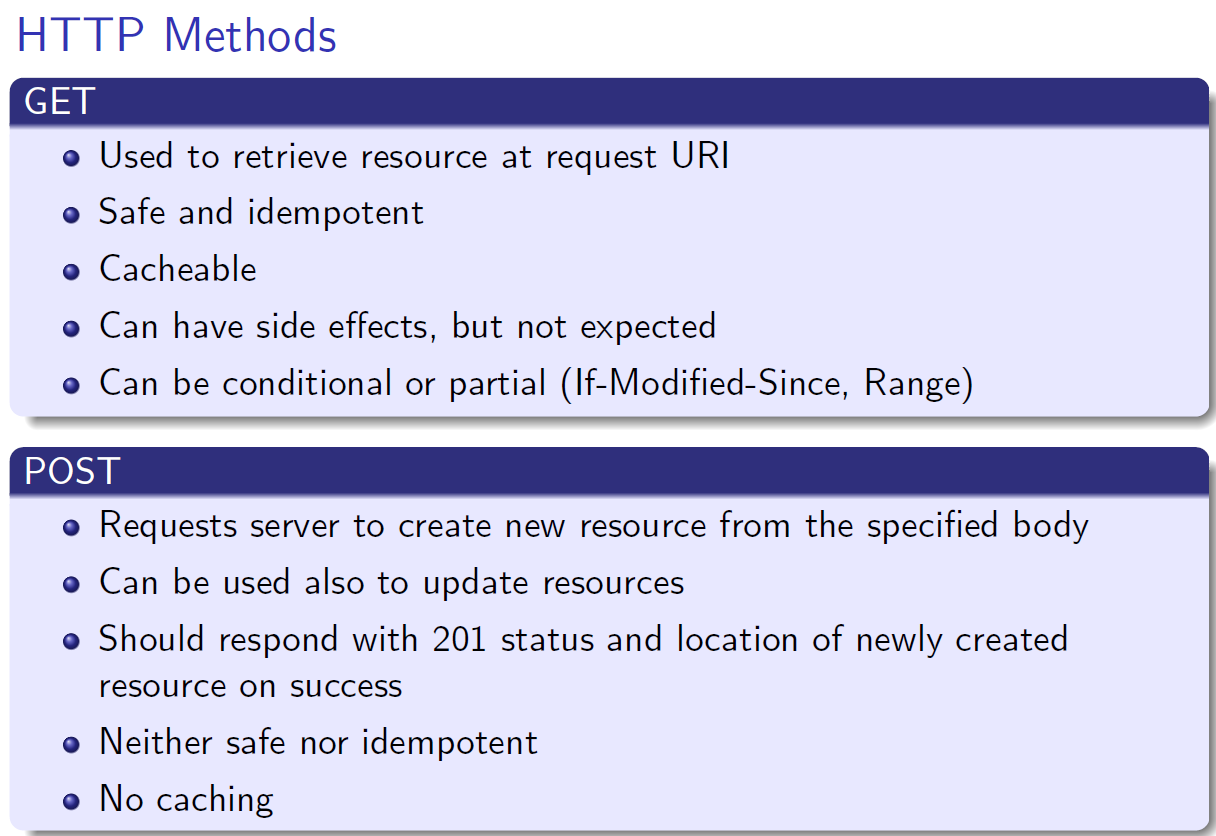




## REST

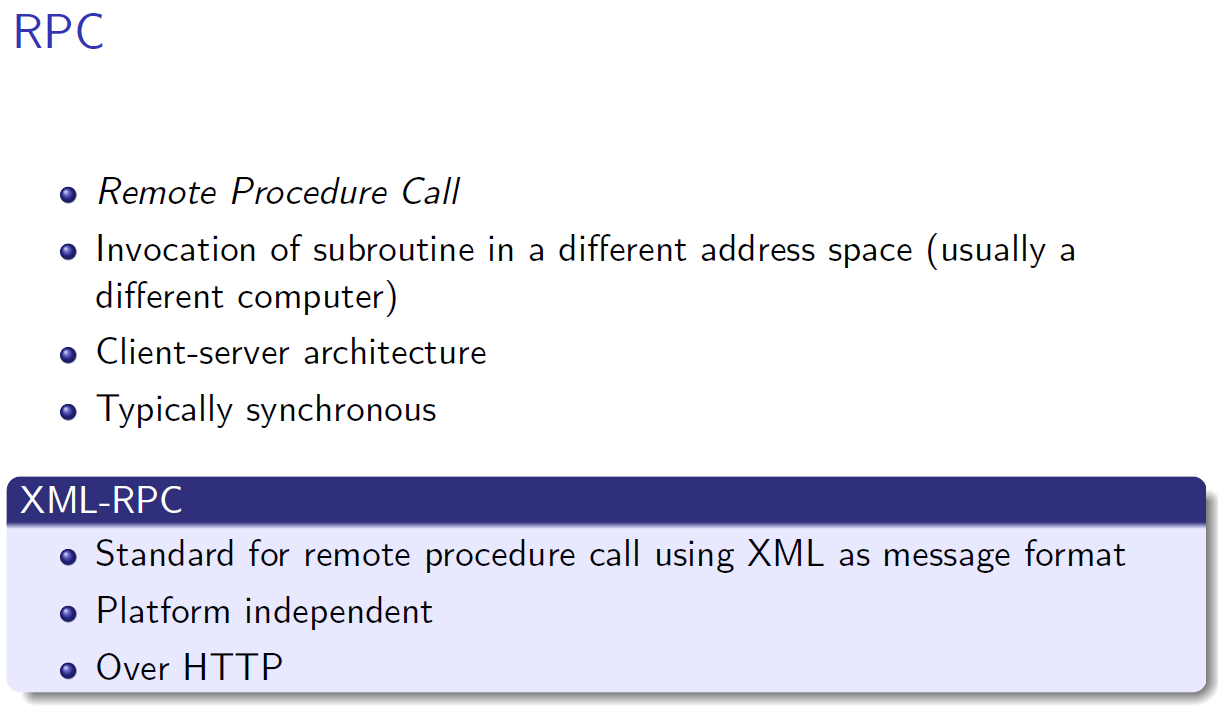


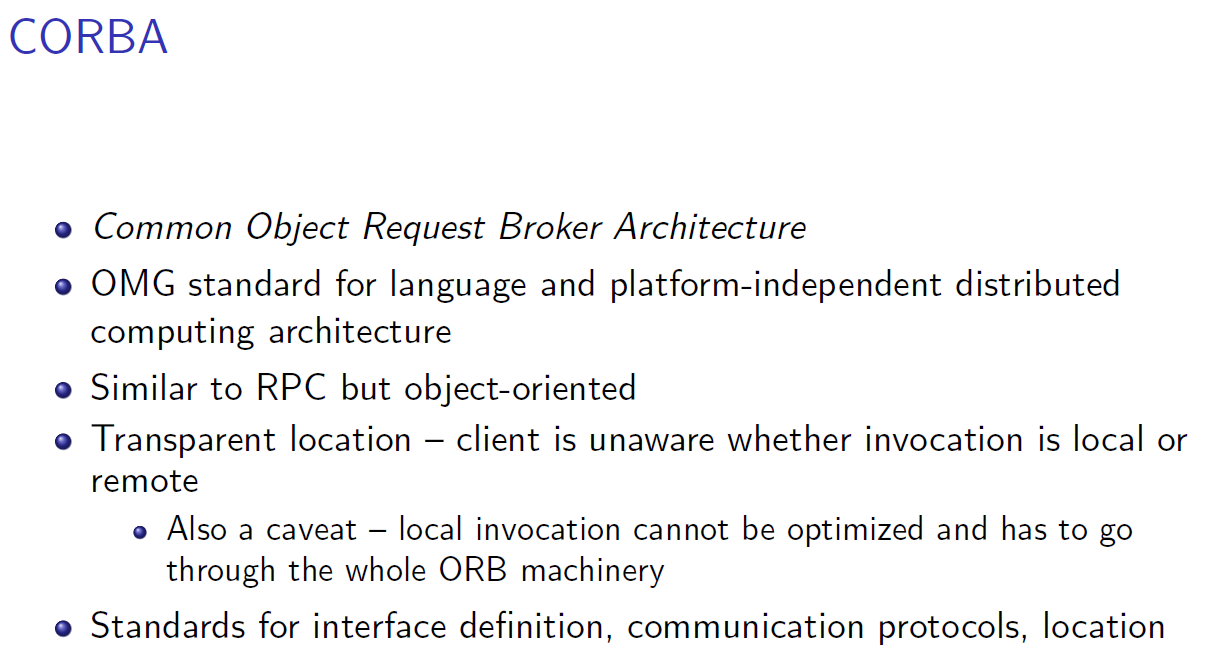


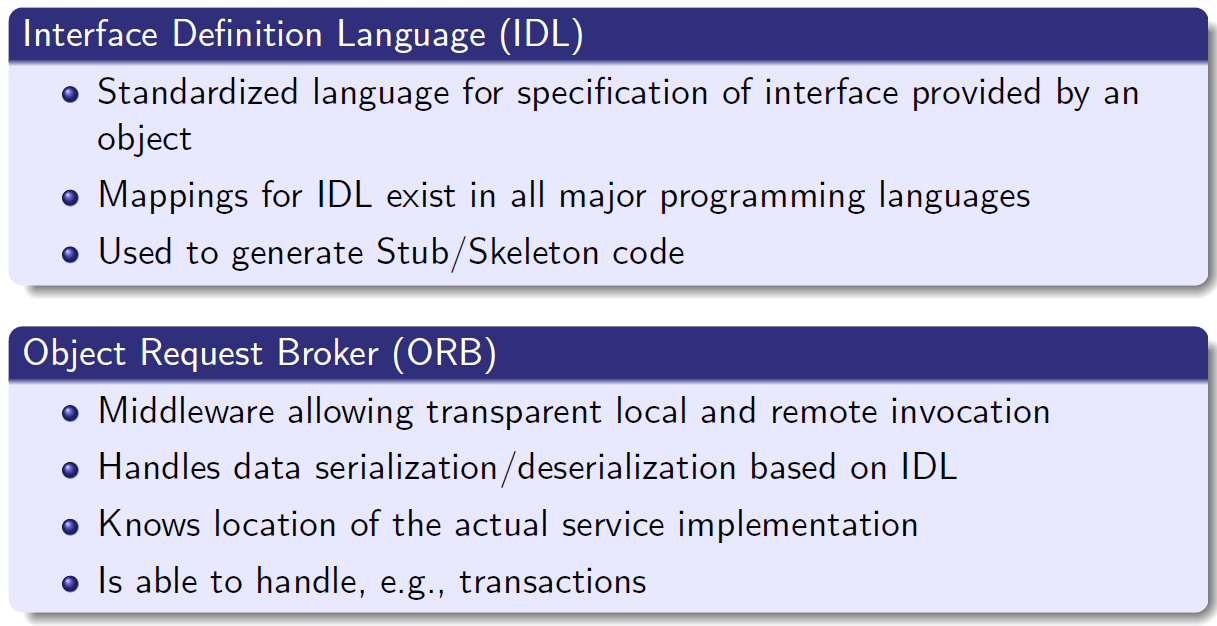
## Linked data

## RPC

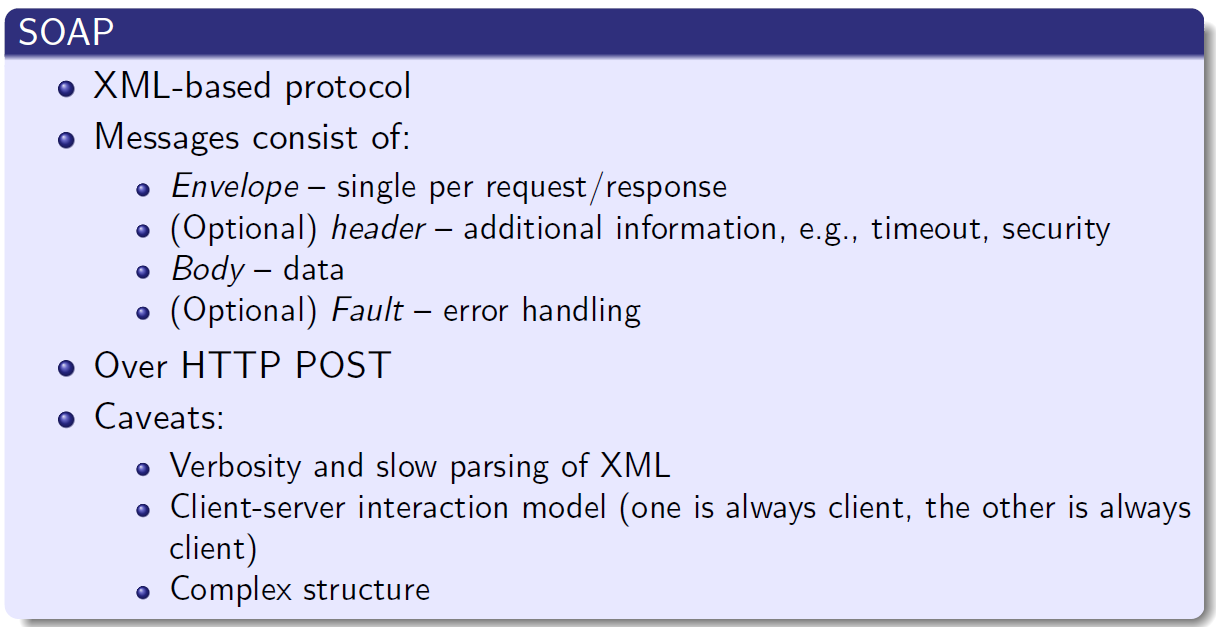
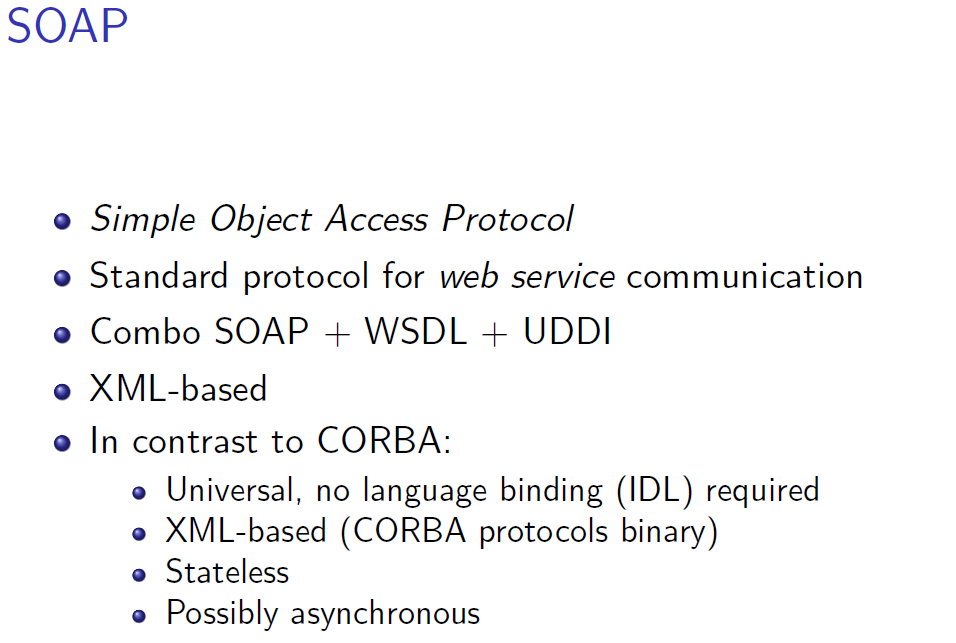


## CORBA

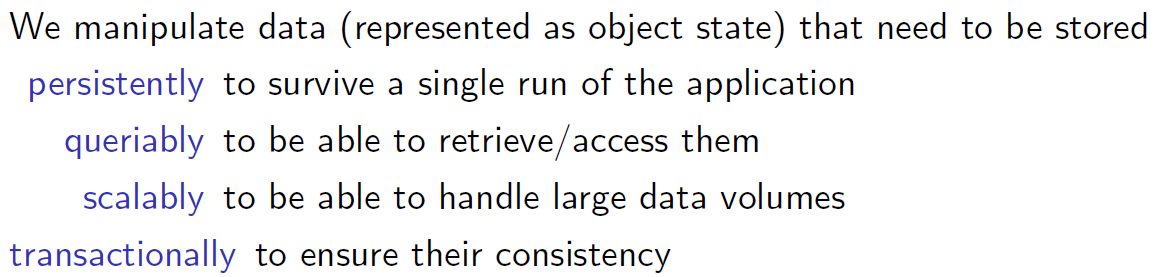


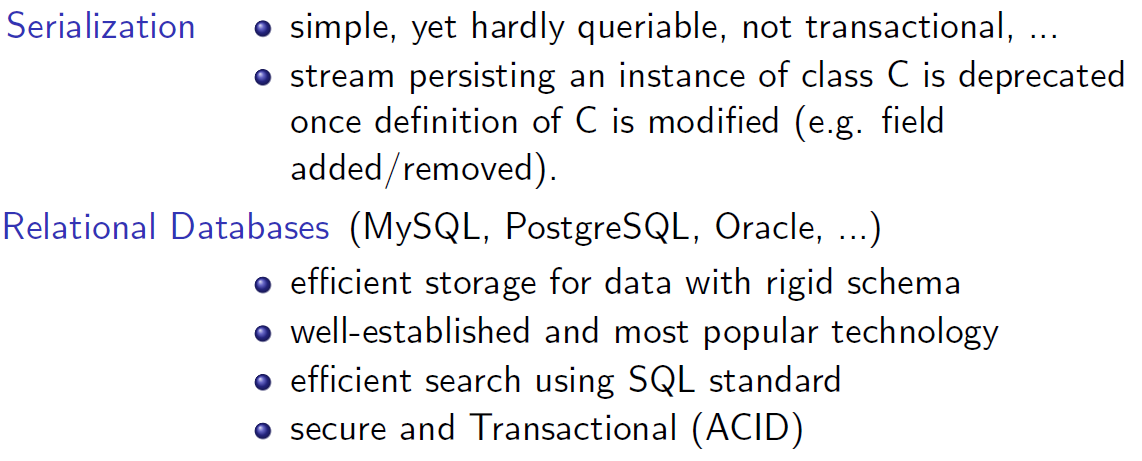


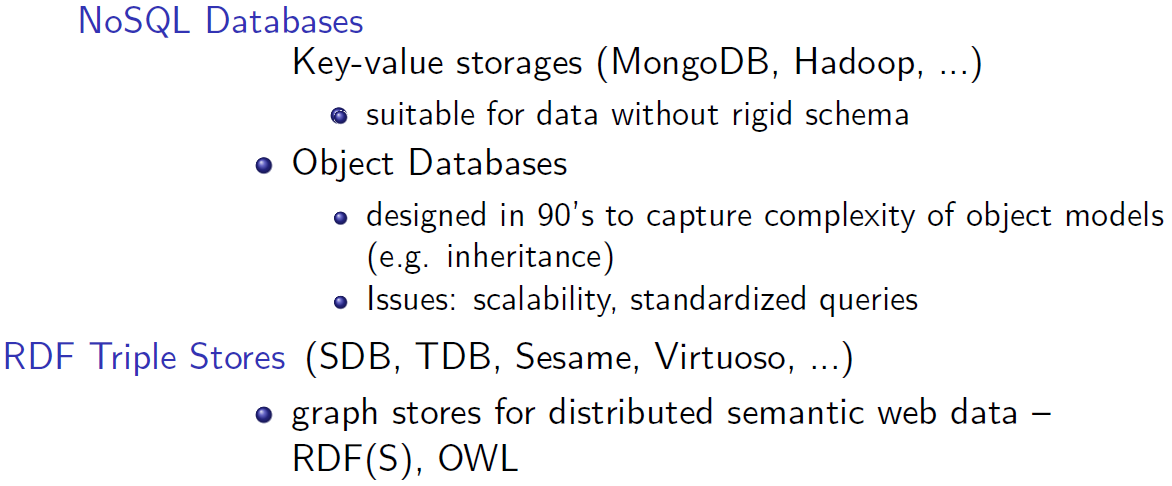
## SOAP



# Persitence





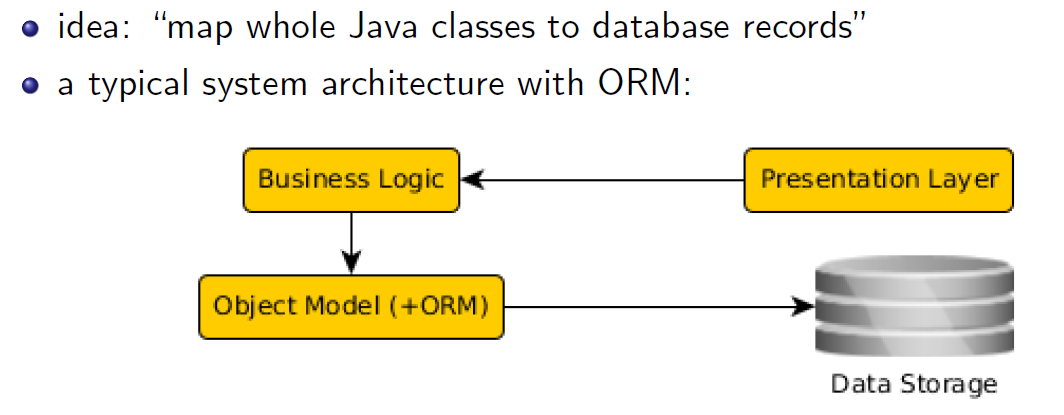


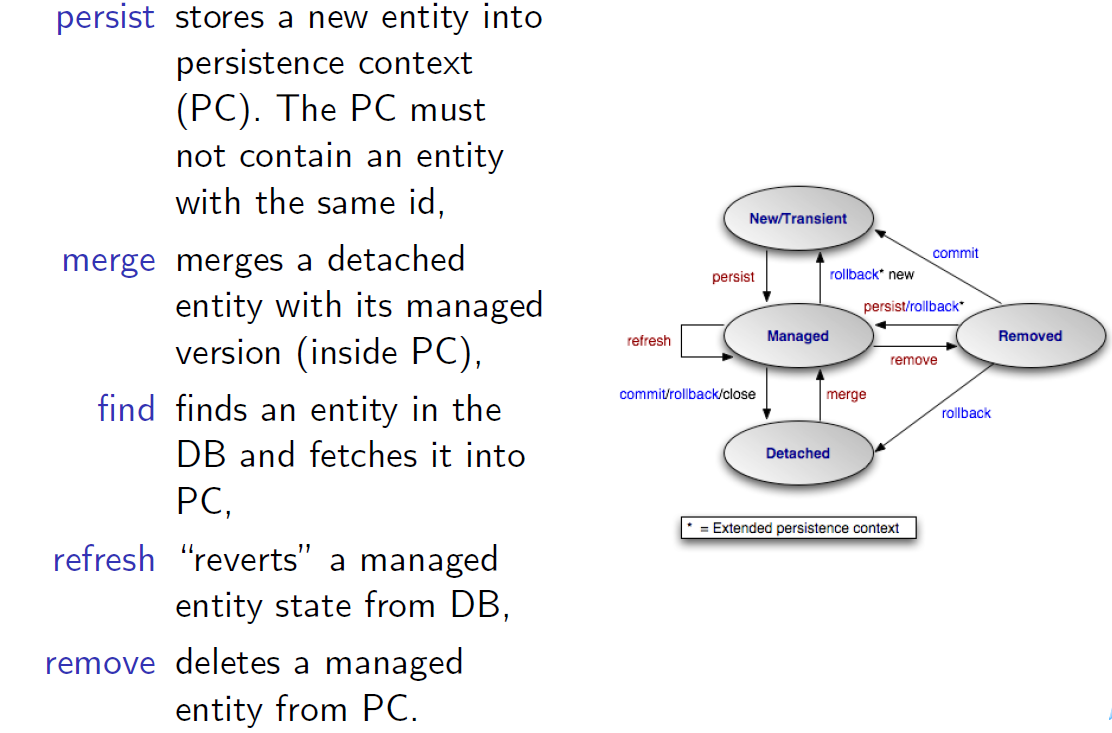
## ORM

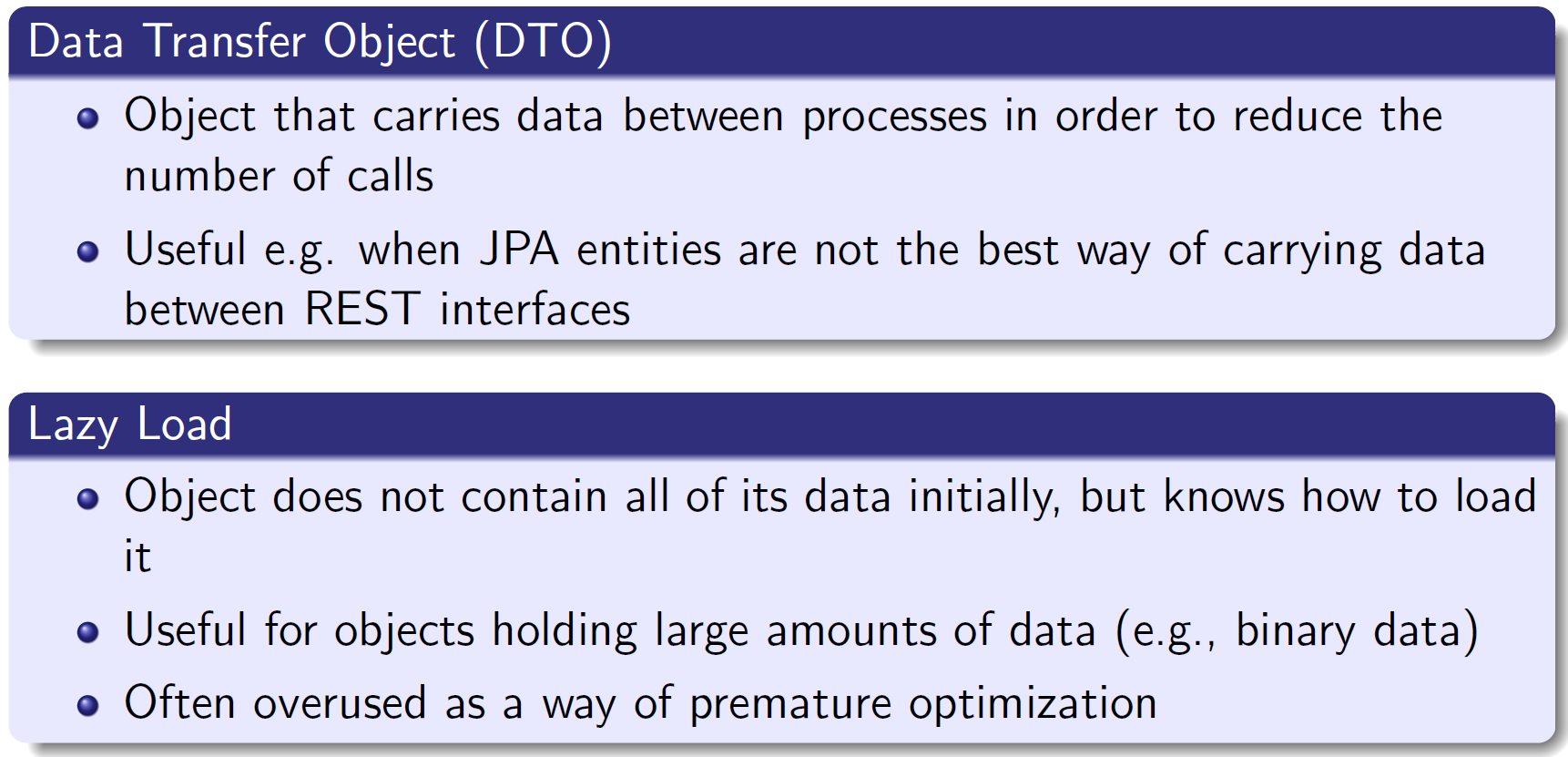
Objektově relační zobrazení (ORM, O/RM nebo O/R zobrazení) je programovací technika v softwarovém inženýrství, která zajišťuje automatickou konverzi dat mezi relační databází a objektově orientovaným programovacím jazykem.

Nicméně hlavním cílem ORM je synchronizace mezi používanými objekty v aplikaci a jejich reprezentací v databázovém systému tak, aby byla zajištěna persistence dat. Vývojář potřebuje persistentně uchovávat objekty, ale nepotřebuje se starat, jak se tato persistence provede.

ORM se dále stará o automatickou konverzi rozdílných datových typů mezi databázovým systémem a programovacím jazykem. Pokročilé techniky ORM také řeší například možnost využití objektové dědičnosti, kterou relační databáze nepodporují.







# JVM Technologie

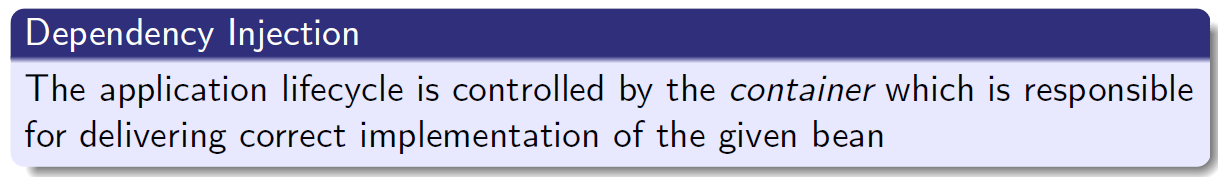
## Java Beans

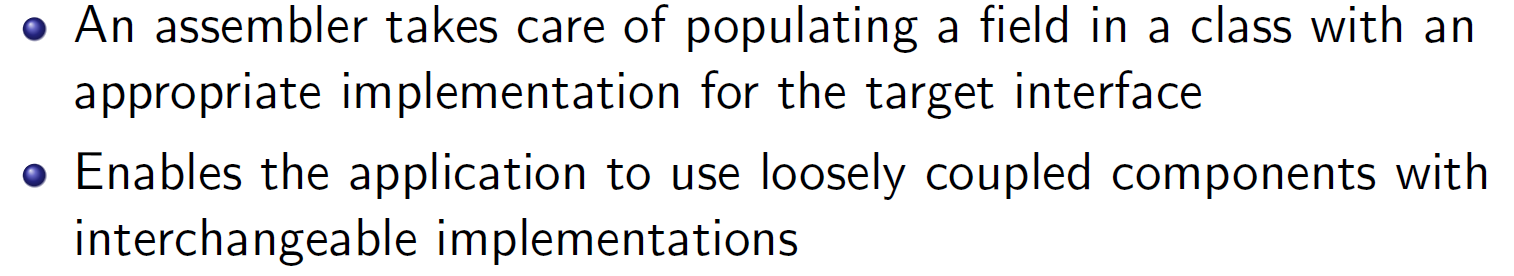
A JavaBean is a Java class that should follow the following conventions:

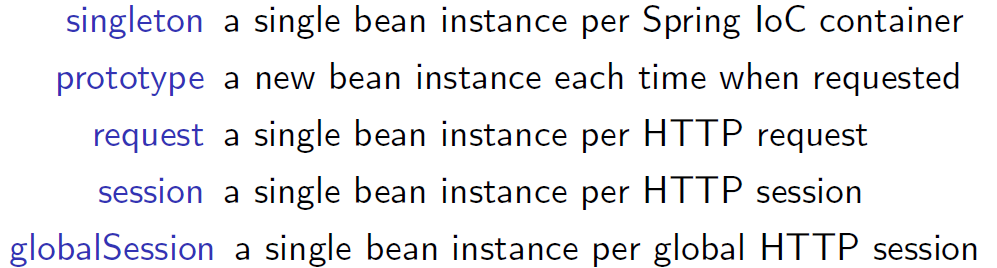
* It should have a no-arg constructor.
* It should be Serializable.
* It should provide methods to set and get the values of the properties, known as getter and setter methods.

In Spring, the objects that form the backbone of your application and that are managed by the Spring IoC container are called beans. A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container.

### Dependency injection

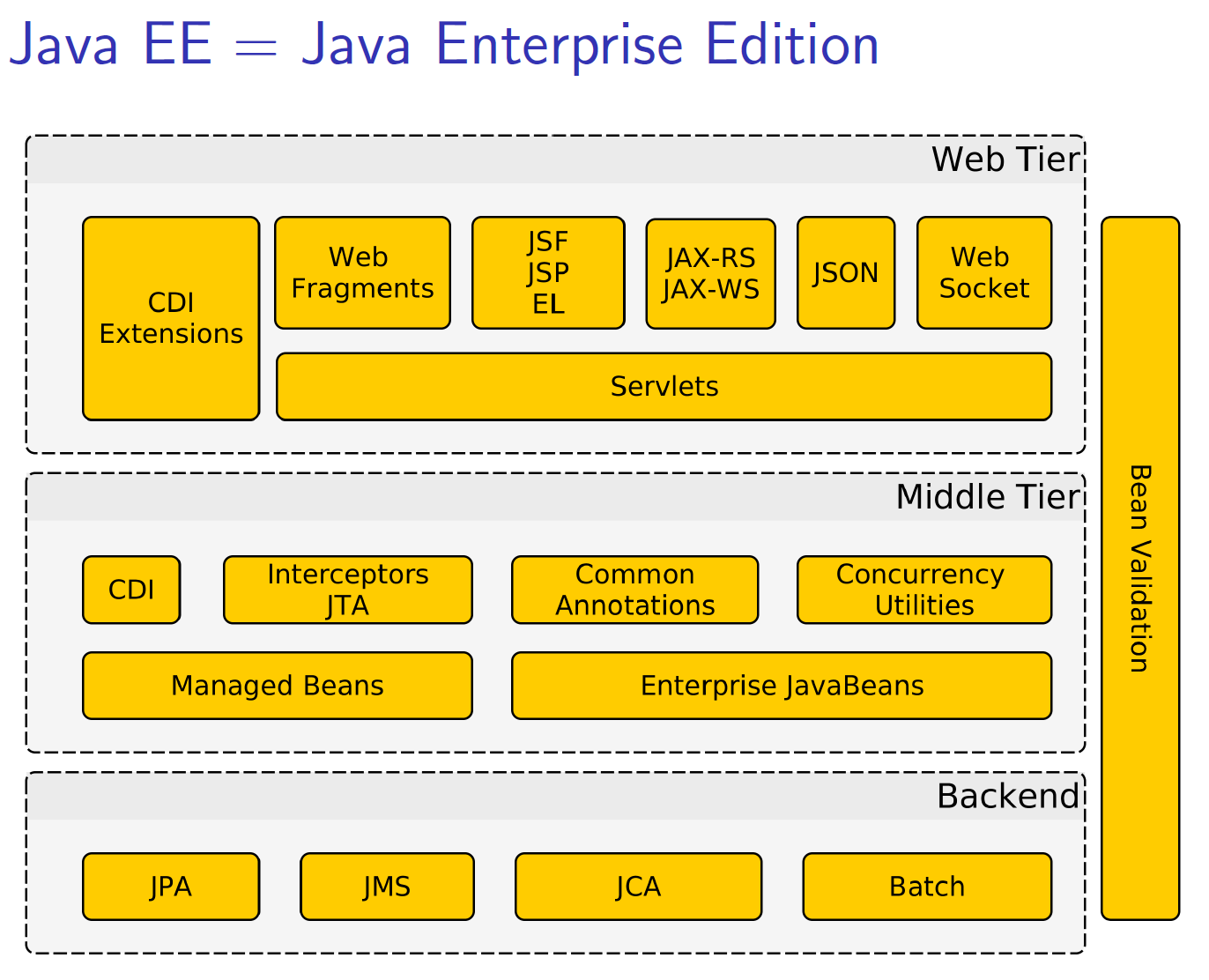
****





## Java EE

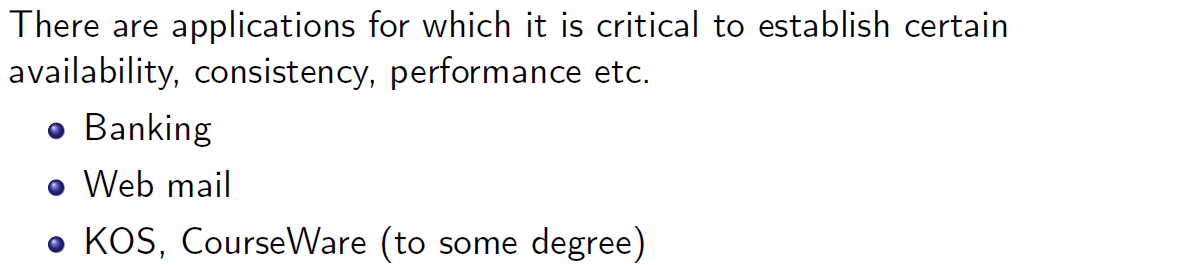
**Java Enterprise Edition** (**Java EE**), formerly **Java 2 Platform, Enterprise Edition** (**J2EE**) is a set of specifications, extending [Java SE](https://en.wikipedia.org/wiki/Java_SE) 8[[1]](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition#cite_note-1) with specifications for enterprise features such as [distributed computing](https://en.wikipedia.org/wiki/Distributed_computing" \o "Distributed computing) and [web services](https://en.wikipedia.org/wiki/Web_service).[[2]](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition#cite_note-2) Java EE applications are run on [reference runtimes](https://en.wikipedia.org/w/index.php?title=Reference_runtime&action=edit&redlink=1), that can be microservices or [application servers](https://en.wikipedia.org/wiki/Application_server" \o "Application server), which handle transactions, security, scalability, [concurrency](https://en.wikipedia.org/wiki/Concurrency_(computer_science)" \o "Concurrency (computer science)) and management of the components it is deploying.

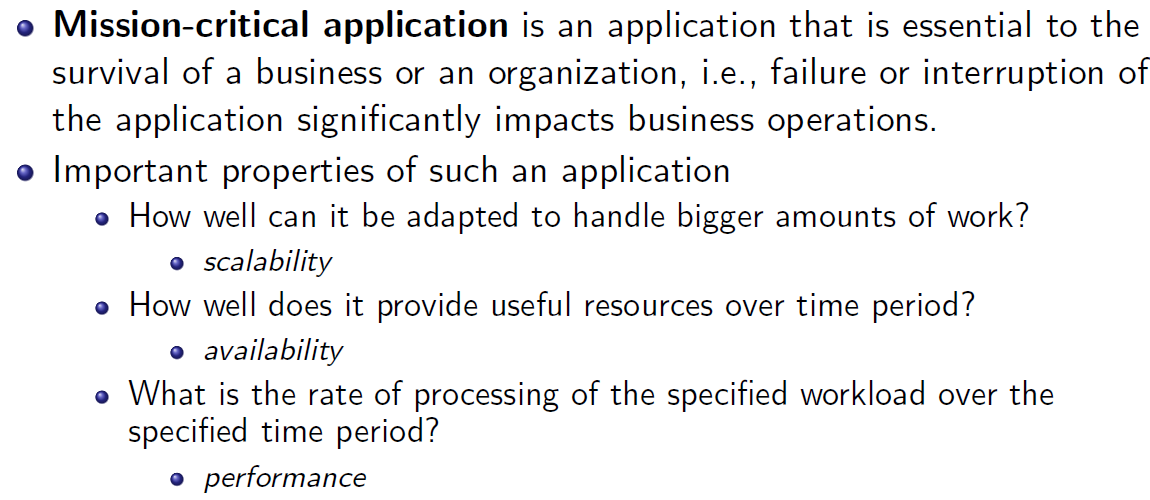


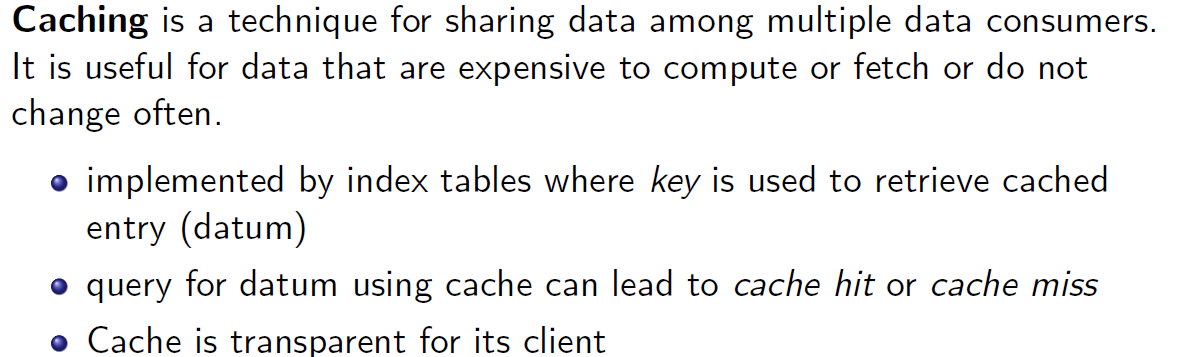
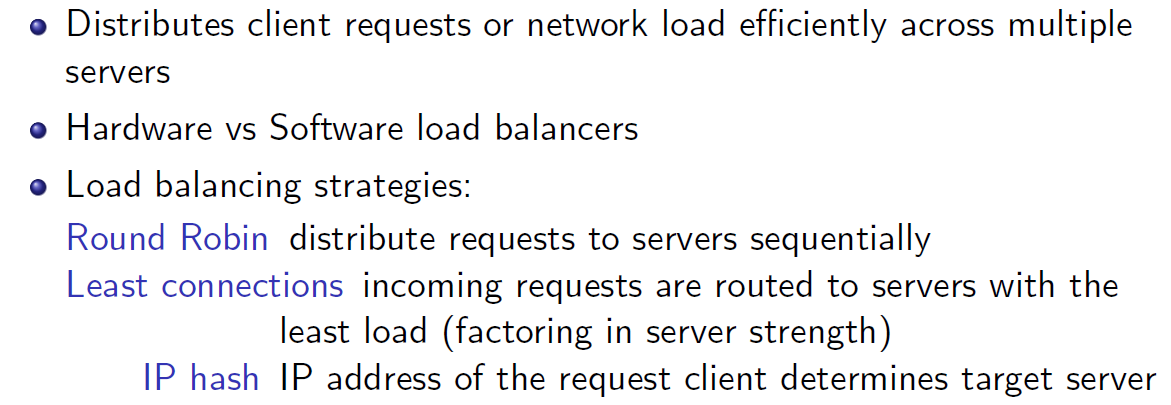
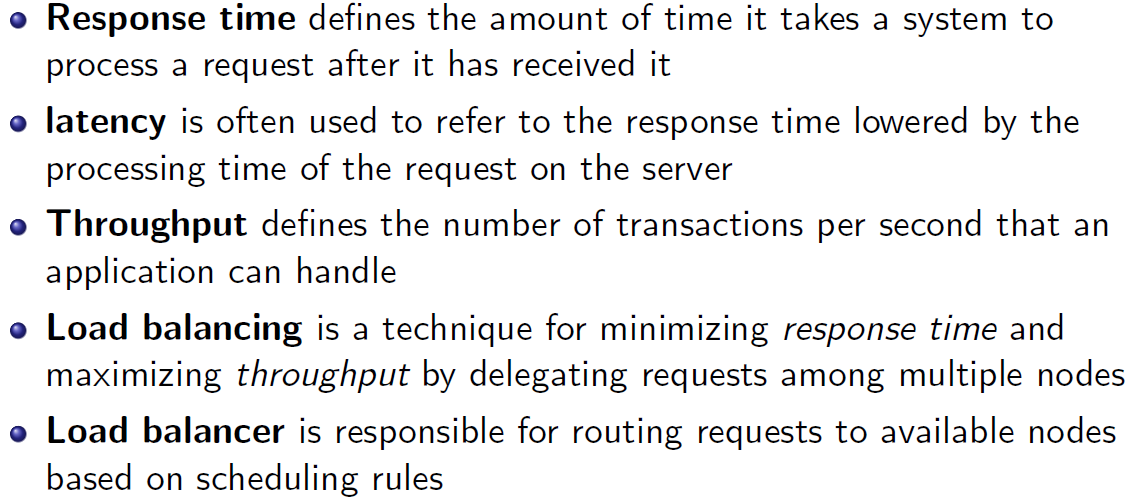
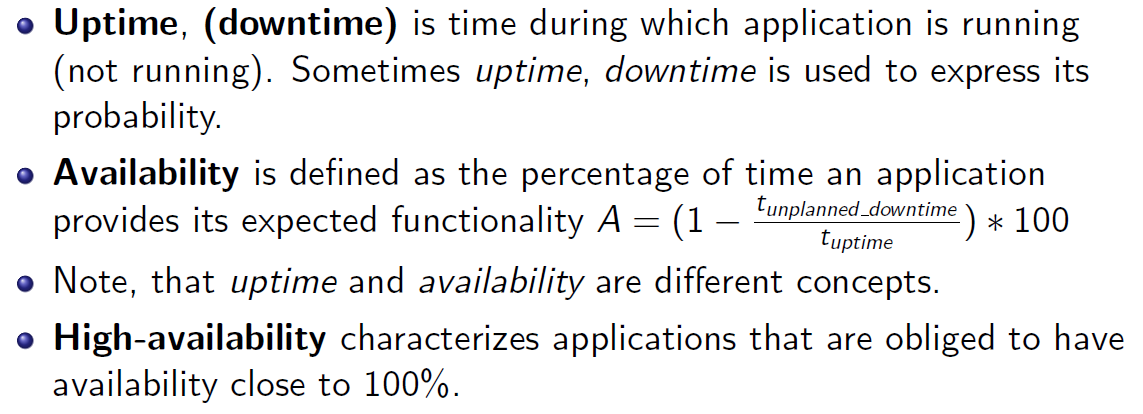
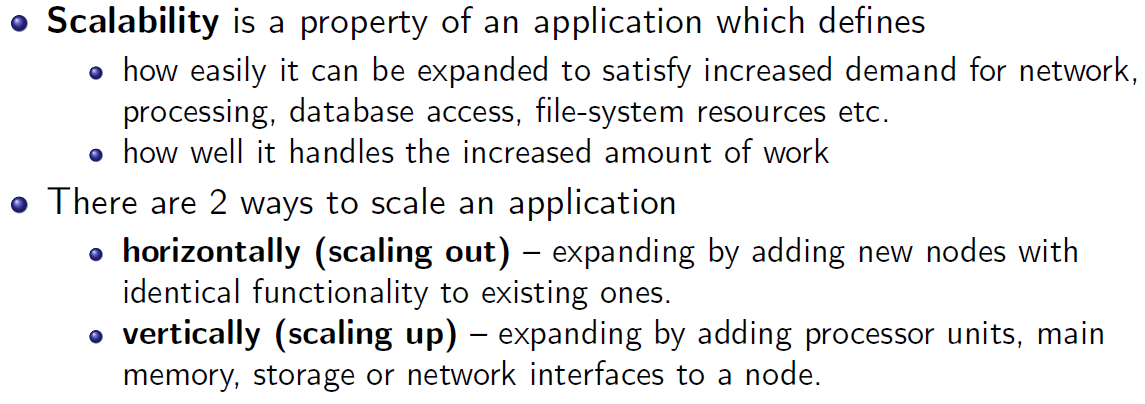
## Spring

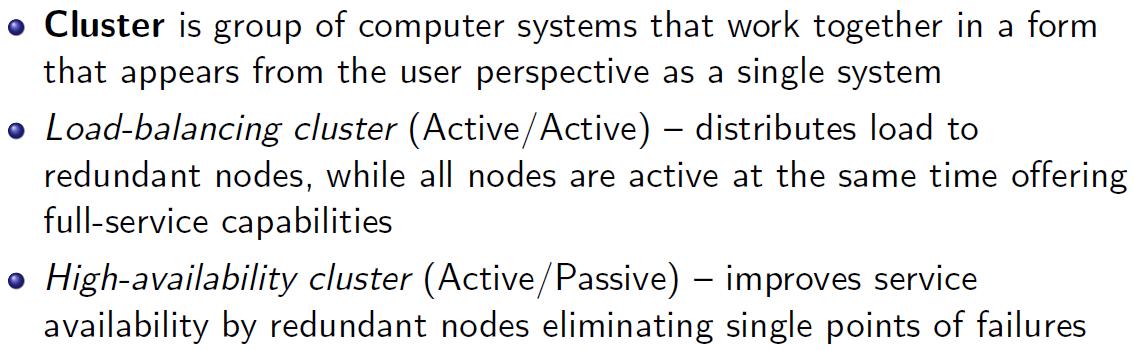
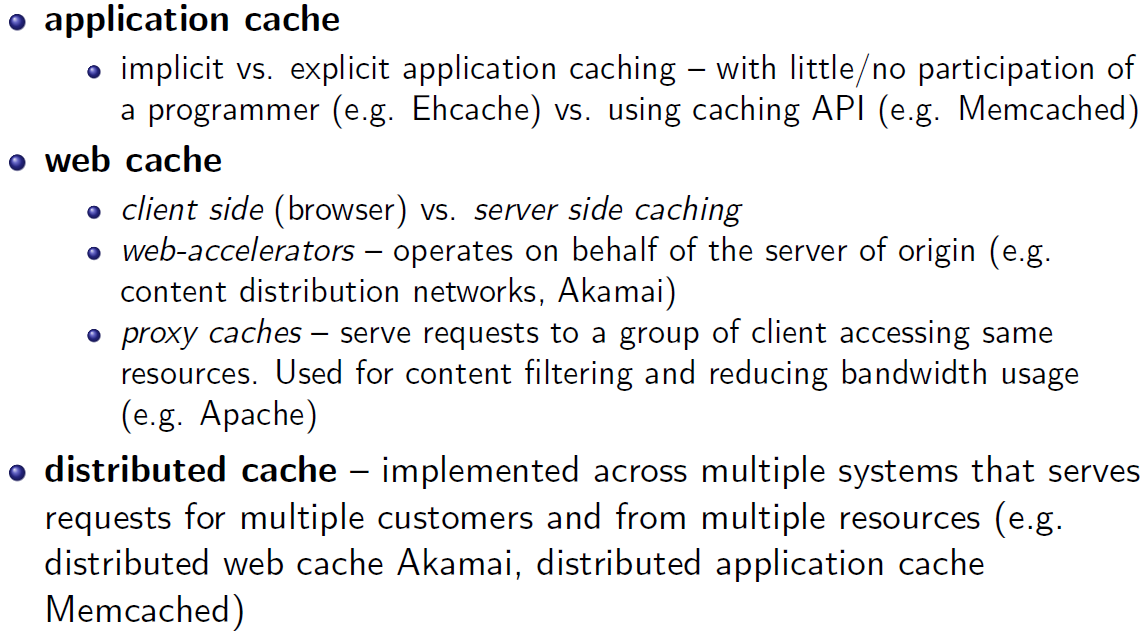
The **Spring Framework** is an [application framework](https://en.wikipedia.org/wiki/Application_framework" \o "Application framework) and [inversion of control](https://en.wikipedia.org/wiki/Inversion_of_control" \o "Inversion of control) [container](https://en.wikipedia.org/wiki/Servlet_container" \o "Servlet container) for the [Java platform](https://en.wikipedia.org/wiki/Java_platform). The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the [Java EE](https://en.wikipedia.org/wiki/Java_EE) (Enterprise Edition) platform. Although the framework does not impose any specific [programming model](https://en.wikipedia.org/wiki/Programming_model" \o "Programming model), it has become popular in the Java community as an addition to, or even replacement for the [Enterprise JavaBeans](https://en.wikipedia.org/wiki/Enterprise_JavaBeans" \o "Enterprise JavaBeans) (EJB) model. The Spring Framework is [open source](https://en.wikipedia.org/wiki/Open-source_software).

# Load Balancing



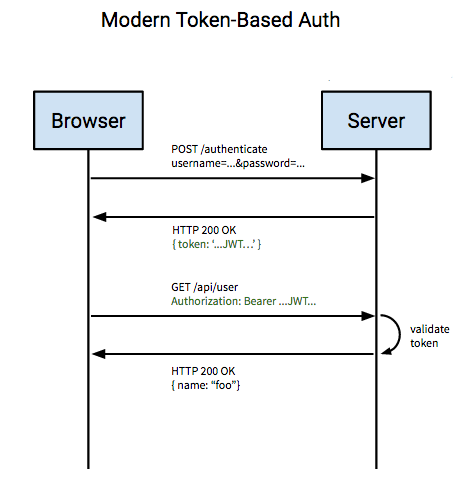




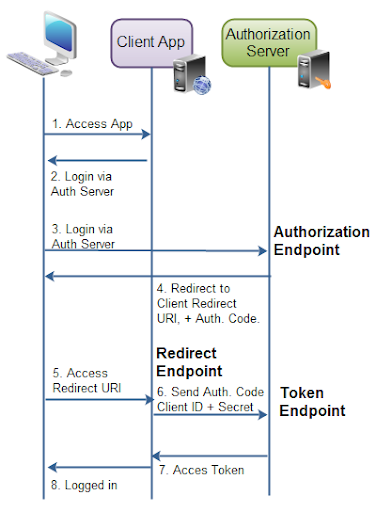


# Zabezpečení

## JWT

**JSON Web Token** (**JWT**, sometimes pronounced [/dʒɒt/](https://en.wikipedia.org/wiki/Help:IPA/English)[[1]](https://en.wikipedia.org/wiki/JSON_Web_Token#cite_note-rfc7519-1)) is an internet standard for creating [JSON](https://en.wikipedia.org/wiki/JSON)-based [access tokens](https://en.wikipedia.org/wiki/Access_token" \o "Access token) that assert some number of claims. The tokens are signed either using a private secret or a public/private key. For example, a server could generate a token that has the claim "logged in as admin" and provide that to a client. The client could then use that token to prove that it is logged in as admin.

## OAUTH 2

**OAuth** is an [open standard](https://en.wikipedia.org/wiki/Open_standard) for access delegation, commonly used as a way for Internet users to grant websites or applications access to their information on other websites but without giving them the passwords.[[1]](https://en.wikipedia.org/wiki/OAuth#cite_note-1) This mechanism is used by companies such

as [Amazon](https://en.wikipedia.org/wiki/Amazon_(company)),[[2]](https://en.wikipedia.org/wiki/OAuth#cite_note-2) [Google](https://en.wikipedia.org/wiki/Google), [Facebook](https://en.wikipedia.org/wiki/Facebook), [Microsoft](https://en.wikipedia.org/wiki/Microsoft) and [Twitter](https://en.wikipedia.org/wiki/Twitter) to permit the users to share information about their accounts with third party applications or websites

Generally, OAuth provides to clients a "secure delegated access" to server resources on behalf of a resource owner. It specifies a process for resource owners to authorize third-party access to their server resources without sharing their credentials. Designed specifically to work with [Hypertext Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP), OAuth essentially allows [access tokens](https://en.wikipedia.org/wiki/Access_token" \o "Access token) to be issued to third-party clients by an authorization server, with the approval of the resource owner. The third party then uses the access token to access the protected resources hosted by the resource server.[[3]](https://en.wikipedia.org/wiki/OAuth#cite_note-3)

## Chyby v zabezpečení

