mobile business

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Lecture 03

Business Informatics 2 (PWIN)
WS 2017/2018

Information Systems II
Models and Architectures

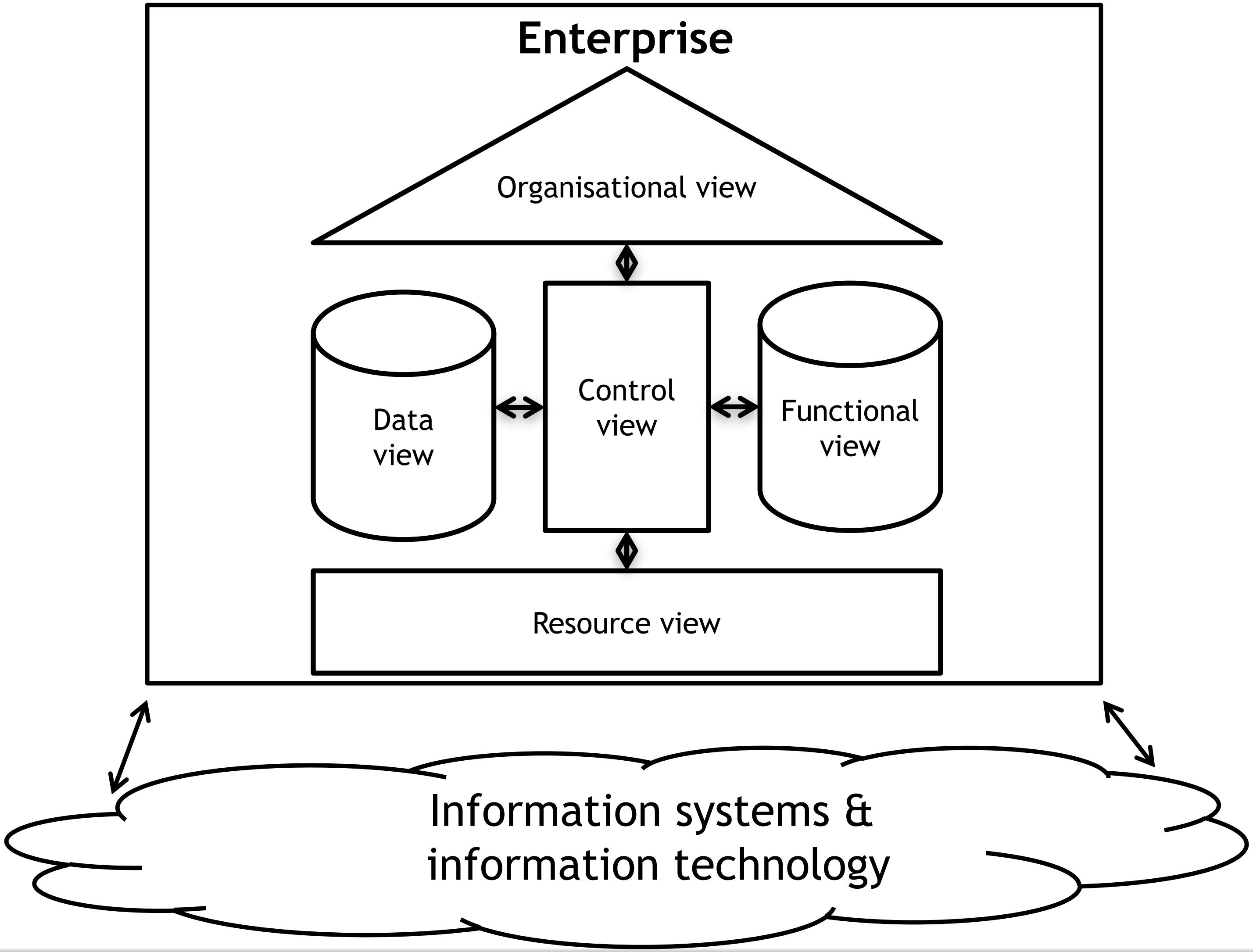
Prof. Dr. Kai Rannenberg

Deutsche Telekom Chair of Mobile Business & Multilateral Security
Johann Wolfgang Goethe University Frankfurt a. M.

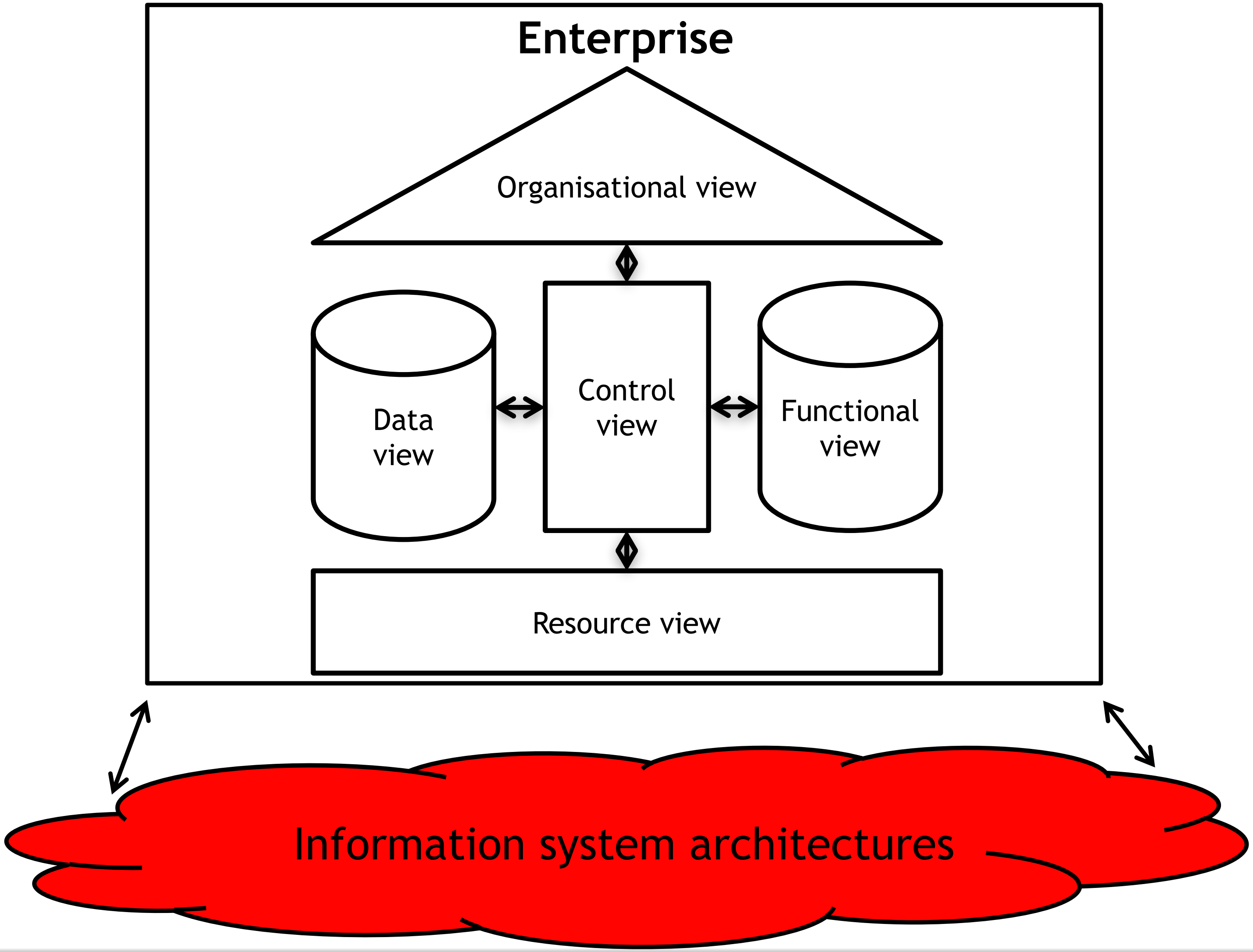
Agenda

- Enterprise Models vs. IS Architecture Models
- Structural Models for IS Architectures
- IS Architecture Concepts

Enterprise Models vs. IS Architecture Models



Enterprise Models vs. IS Architecture Models



Agenda

- Enterprise Models vs. IS Architecture Models
- Structural Models for IS Architectures
- IS Architecture Concepts

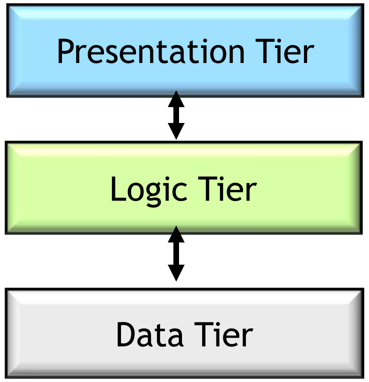
Requirements for the Structure of IS Architectures

- Minimisation of complexity for IS Components
- Scalability of IS components
- Portability of IS components
- Maintainability of IS components
- Standardisation of IS components
- Well-defined interfaces between IS components
- Independence of IS components

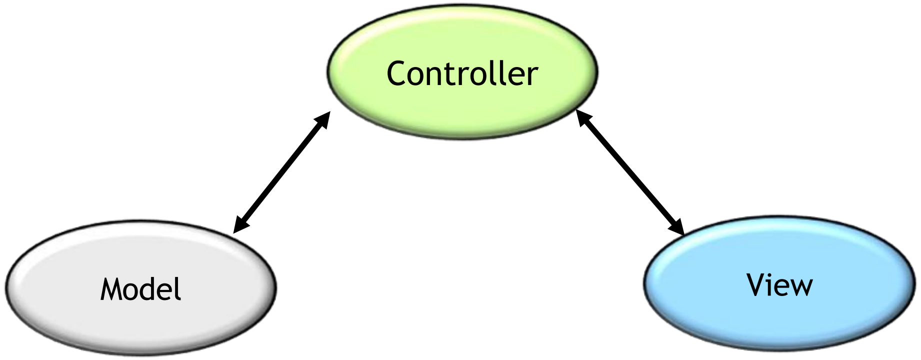
Modularisation of IS components

Two Common Structural Models for IS Architectures

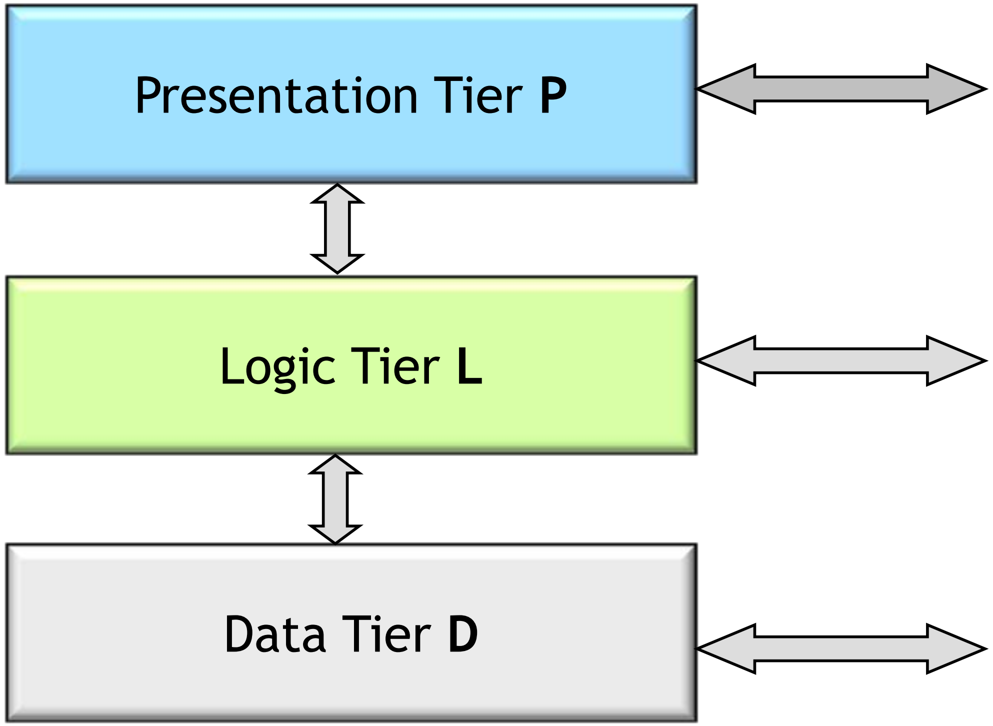
- Three-tier concept



- Model-view-controller (MVC) concept



Three-Tier Concept



Presentation Tier **P**

Logic Tier **L**

Data Tier **D**

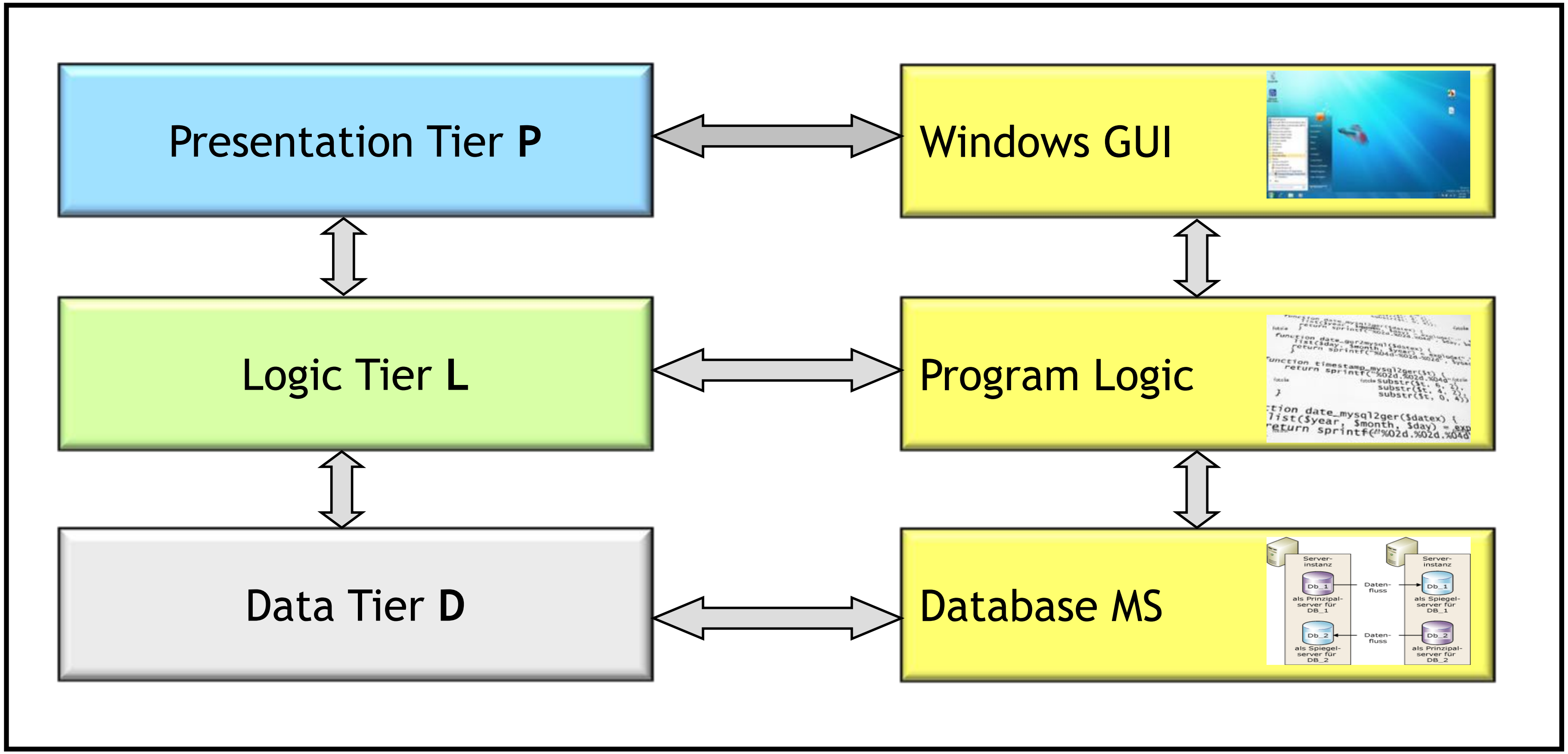
Receives user input / display of data to the user (GUI)

Contains all the program logic

Stores and manage the data

Three-Tier Concept Example (1)

Conventional IS



Presentation Tier **P**

Windows GUI

Logic Tier **L**

Program Logic

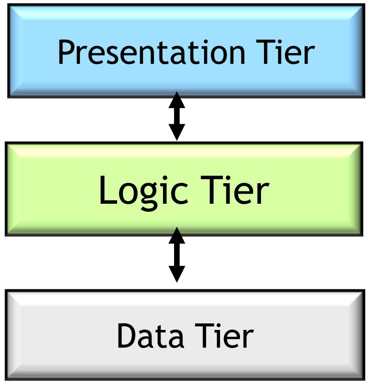
```
function date_ger2mysql($datex) {
    list($day, $month, $year) = explode(
    return sprintf("%04d-%02d-%02d", $year
}
function timestamp_mysql2ger($t) {
    return sprintf("%02d.%02d.%04d",
                    substr($t, 6, 2),
                    substr($t, 4, 2),
                    substr($t, 0, 4));
}
ction date_mysql2ger($datex) {
list($year, $month, $day) = exp
return sprintf("%02d.%02d.%04d"
```

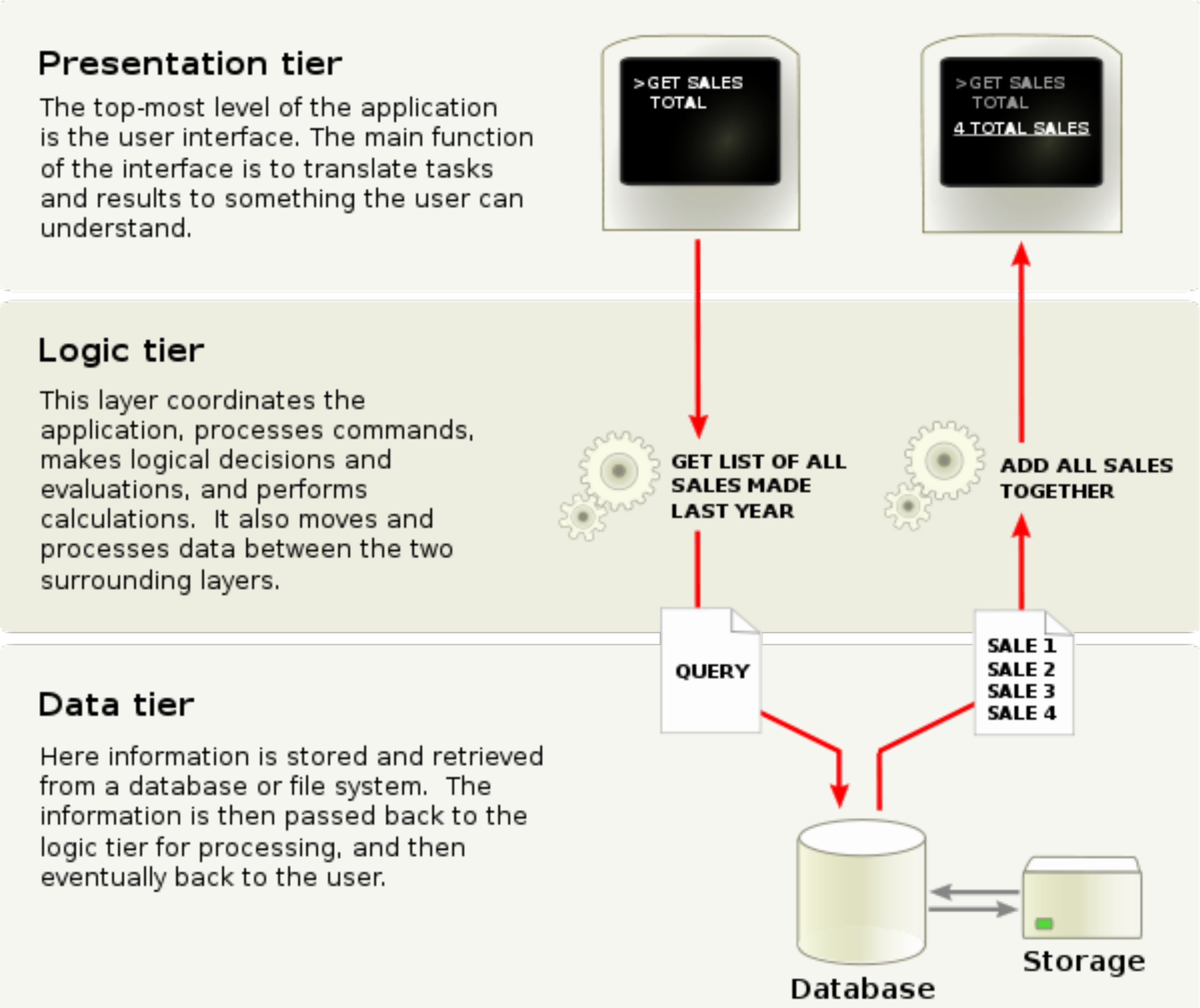
Data Tier **D**

Database MS

Server-instanz
Db_1
als Prinzipal-server für DB_1
Db_2
als Spiegel-server für DB_2
Daten-fluss
Daten-fluss
Server-instanz
Db_1
als Spiegel-server für DB_1
Db_2
als Prinzipal-server für DB_2

Three-Tier Concept Example (2)





Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

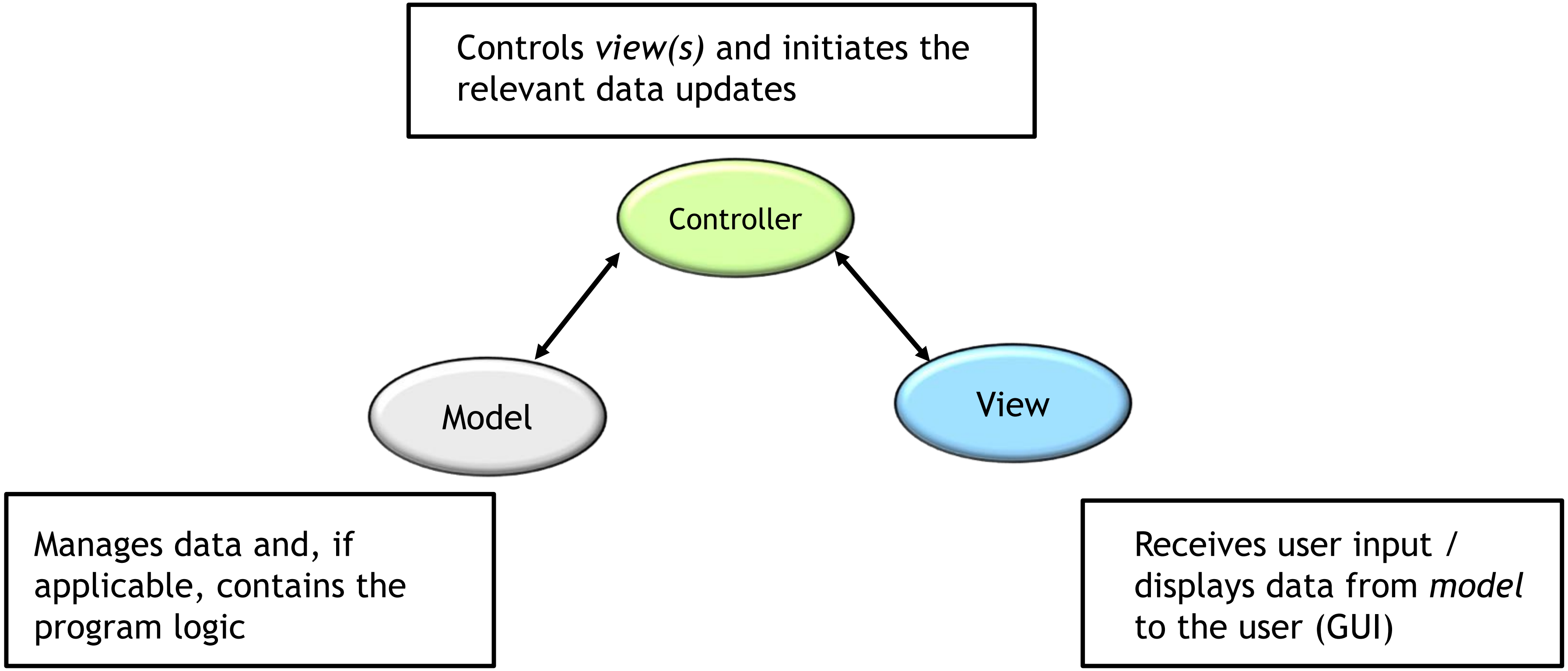
This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Source: Wiki Commons, 2011

Model-View-Controller Concept

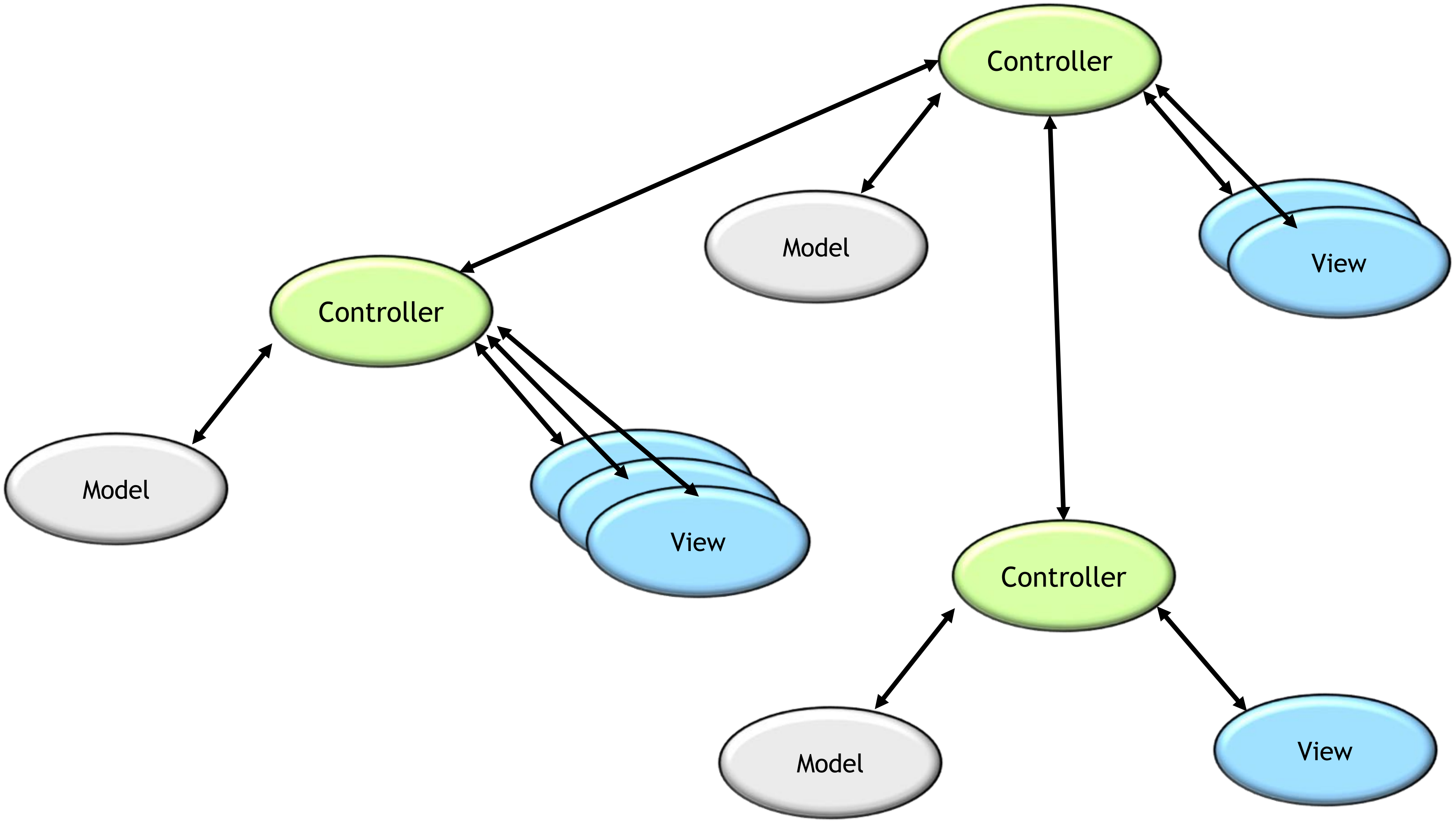


Controls *view(s)* and initiates the relevant data updates

Manages data and, if applicable, contains the program logic

Receives user input / displays data from *model* to the user (GUI)

More Complex MVC Concept



Summary on Three-Tier and MVC Concept

- Similar concepts for structuring IS architectures

- Neither one of the concepts is universally defined or specified, e.g.
  - Two-tier concepts are also in existence (two-tier architecture)
  - Program logic resides sometimes in the *model* and other times in the *controller* (MVC architecture)

- **In conclusion:**

  Independent of the underlying structural models for IS architectures, make sure to modularise certain categories of functionality in an IS.

Agenda

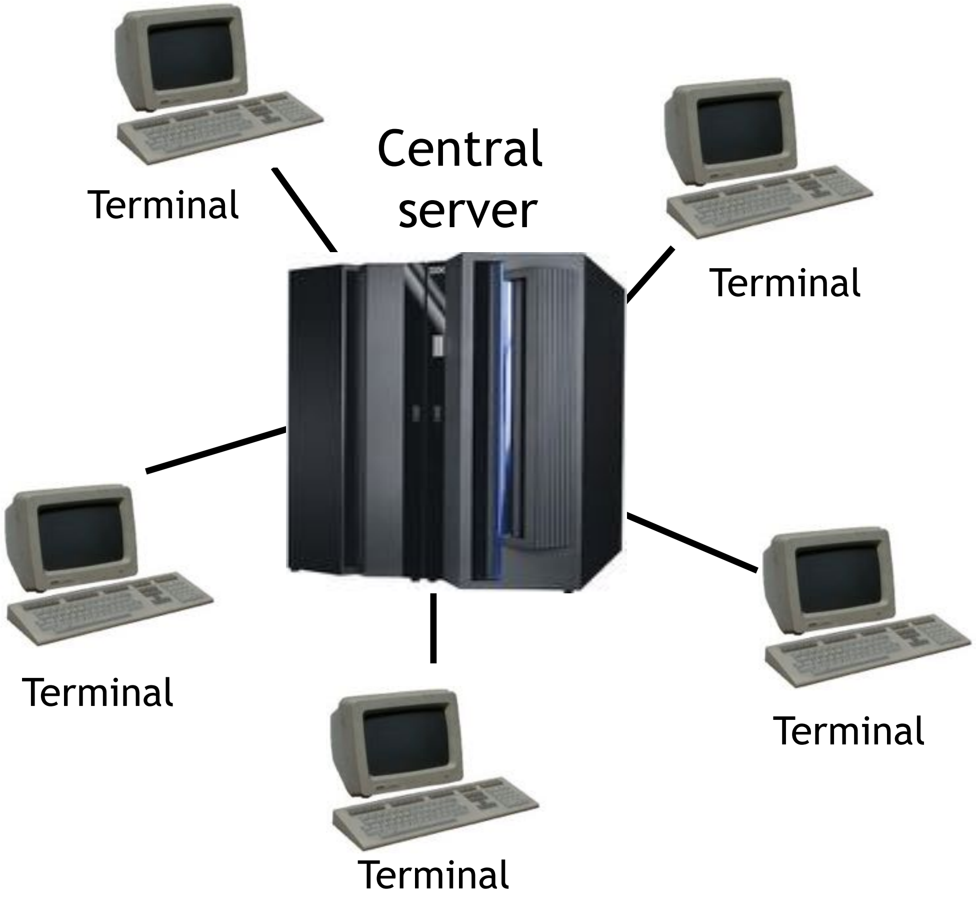
- Enterprise Models vs. IS Architecture Models
- Structural Models for IS Architectures
- IS Architecture Concepts

Architecture Concepts of Networked IS

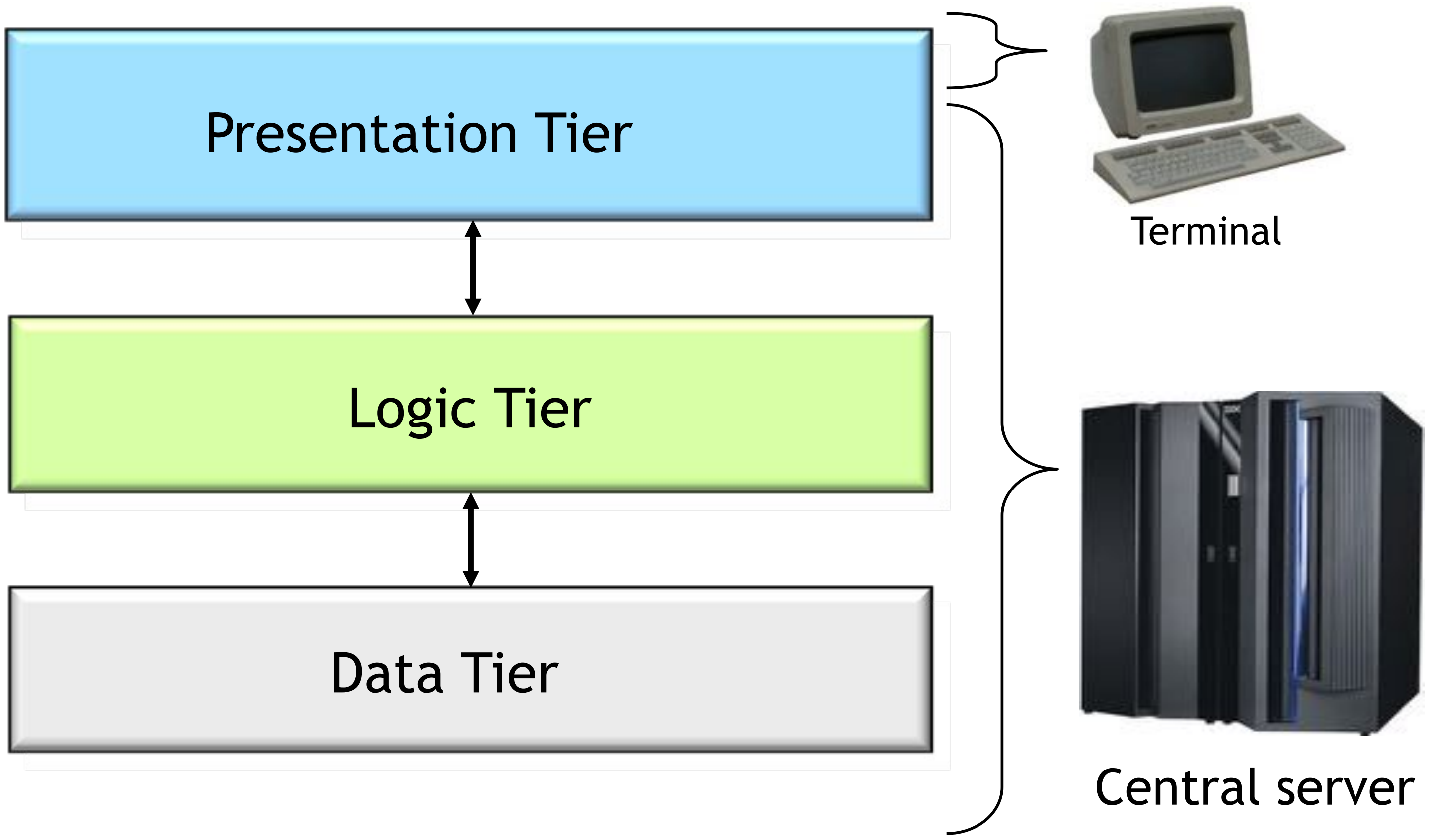
- Central Server Architecture
  Low-feature terminals (receiver of services) attached to a powerful central computing unit (provider of services)
- Client / Server Architecture
  Network of computers, which can take the role of a server (provider of services), a client (receiver of services) or both.
- Cloud Computing Architecture
  Network of computers in the role of a client (receiver of services) connected to a "cloud" of computers (provider of services), which act as a single central server
- Peer-to-Peer Architecture
  Network of computers holding equal rights (provider / receiver of services)
- Edge Computing Architecture
  Leverages network resources to optimise cloud computing systems by performing data processing at the edge of the network, near the data source

Central Server Architecture

- One powerful central computer
- „Dumb“ low-feature terminals (often even without hard drive)
- Terminals provide only the graphical user interface (GUI)
- Central server in charge of processing applications
- Central server takes care of database and its management



Central Server Concept Along the Structural Three-Tier Architecture



Review of the Central Server Architecture Concept

- Advantages
  - Central, common data storage
  - Homogenous application environment
  - No terminal administration required
  - Low-cost terminals

- Disadvantages
  - Single point of failure
  - Fixed network structure
  - Monolithic
  - Cost-intensive central servers
  - Problematic in case of huge traffic and amounts of data

Industry Central Server Solutions

Hardware

FUJITSU COMPUTERS SIEMENS

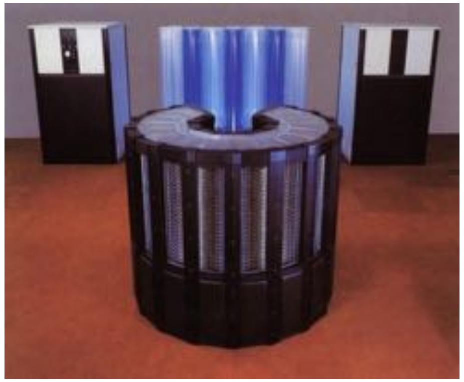
Sun microsystems

take it to the nth

hp

invent

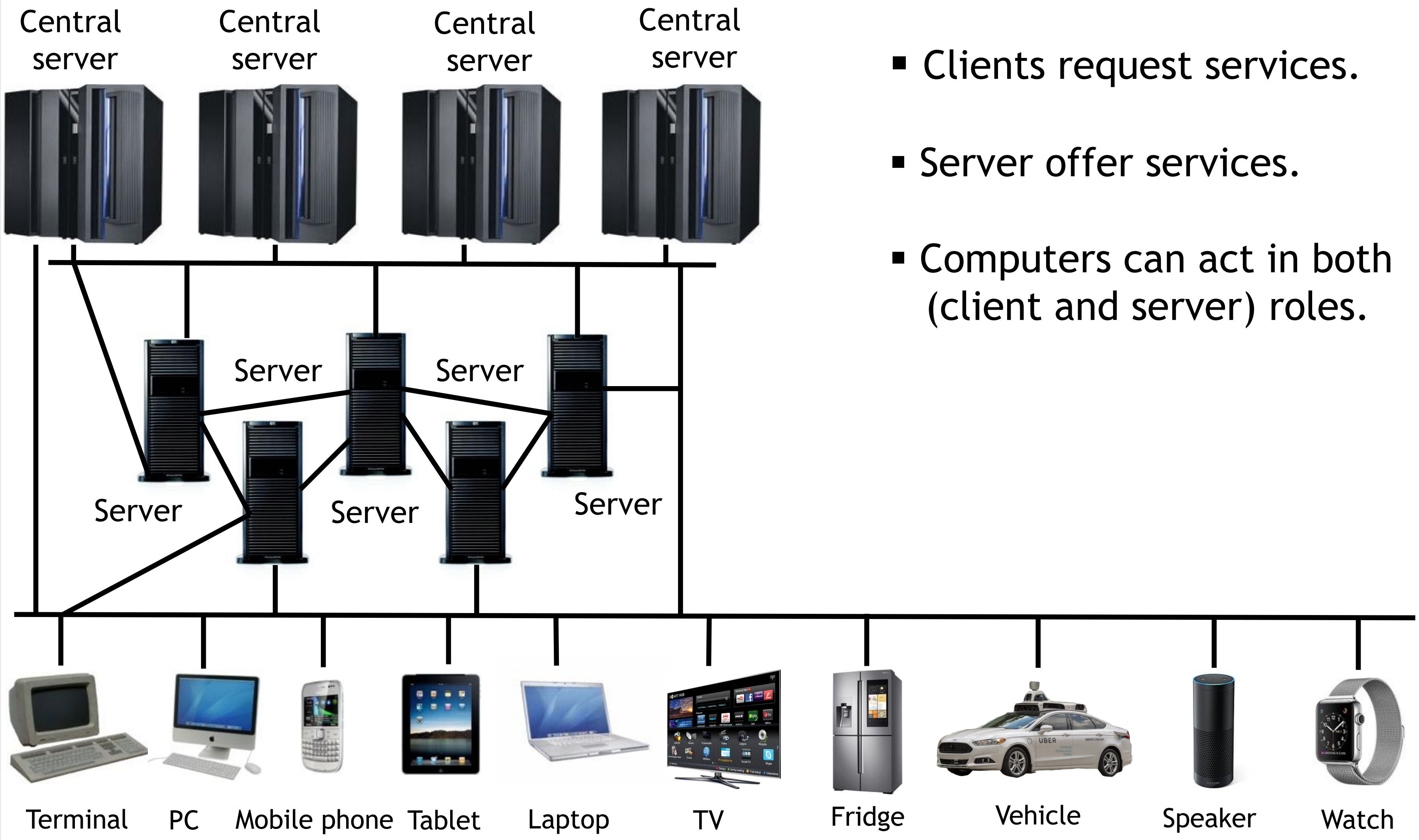
IBM



Operating Systems

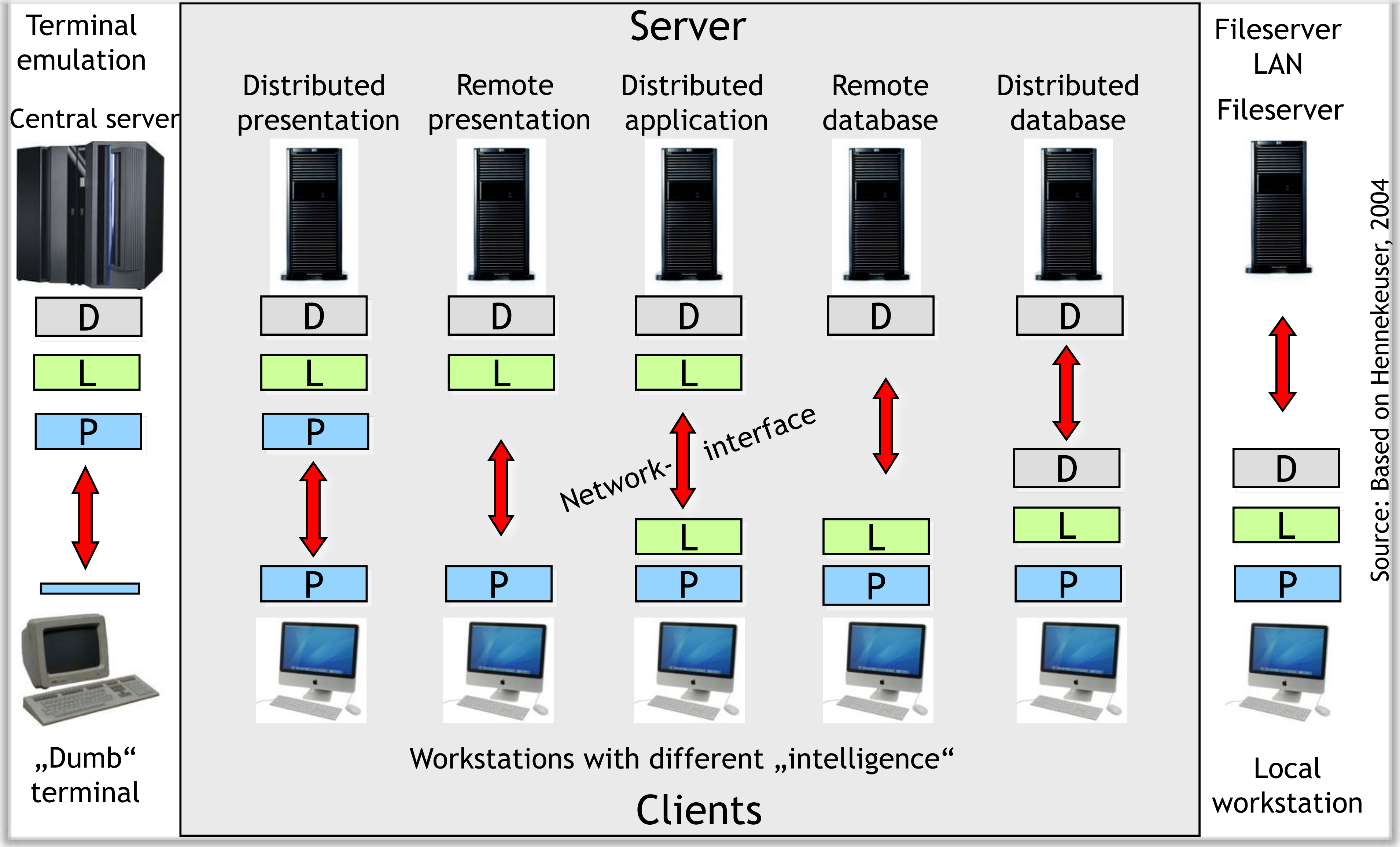
- Unix
- BS 2000
- OS/390
- MVS
- z/OS
- …

Client/Server Architecture



- Clients request services.
- Server offer services.
- Computers can act in both (client and server) roles.

Client/Server Architecture Along the Three-Tier Structural Concept

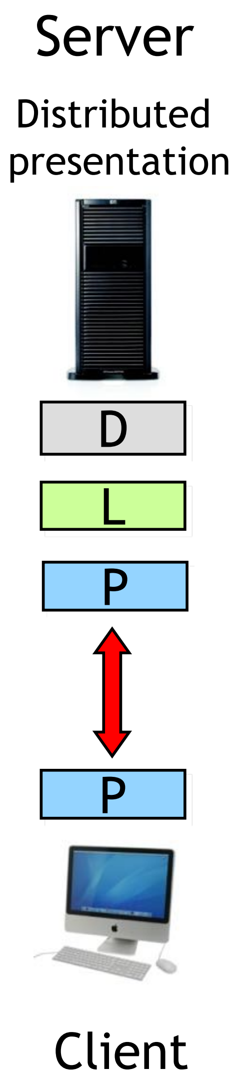


Source: Based on Hennekeuser, 2004

Distributed Presentation

Division of the presentation between server and client:

- Abstract part of the presentation (server)
  Objects (e.g. a window) are created in an abstract manner, i.e. without any concrete representation and functionality.
- Platform-specific part of the presentation (client)
  Abstract objects are created and represented in a platform-specific manner (e.g .making use of the platform’s GUI).
- Advantages of this approach
  Heterogeneous application systems can be integrated into a unified user interface or used on different platforms.
- Examples:
  - X-Windows: A user interface using X-Windows can be represented on multiple platforms.
  - Mobile Web App within Native App: Spiegel Online



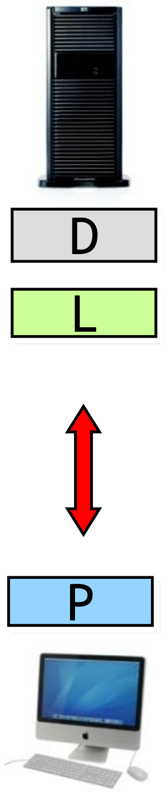
Remote Presentation

Presentation is outsourced to the client:

- Outsourcing of the presentation to the client is especially beneficial, if the central server has no own user interface.
- Clients are able to run on several different platforms.
- User interfaces can be individually customised according to users' needs (e.g. GUI).
- Client can not be a „dumb“ terminal.
- Examples:
  - Citrix XenDesktop
  - TeamViewer
  - Apple Airplay

Server

Remote presentation



D

L

P

Client

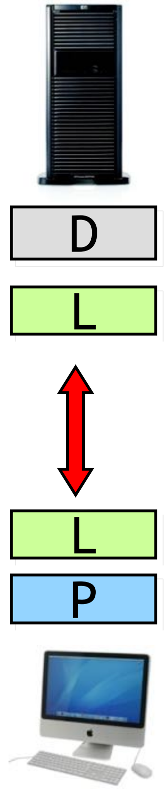
Distributed Application

Division of the application functions (logic) between server and client:

- Centrally used application functions are hosted on the server in order to be available for everyone.
- Decentralised applications reside on the respective client.
- Central application functions will only be used on demand.
- Advantages: Development and maintenance of application functions get simplified; complexity is reduced.
- Examples:
  - Groupware
  - Facebook App
  - DB Navigator App
  - Siri

Server

Distributed application

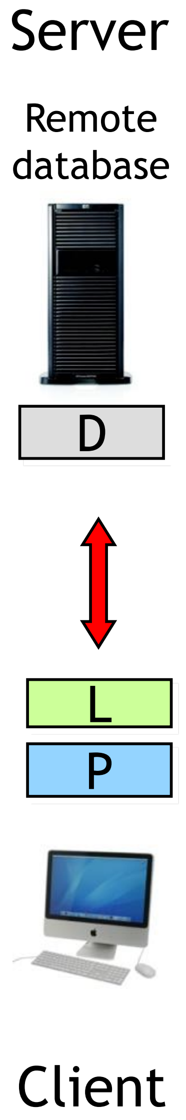


Client

Remote Database

Data management resides on the server:

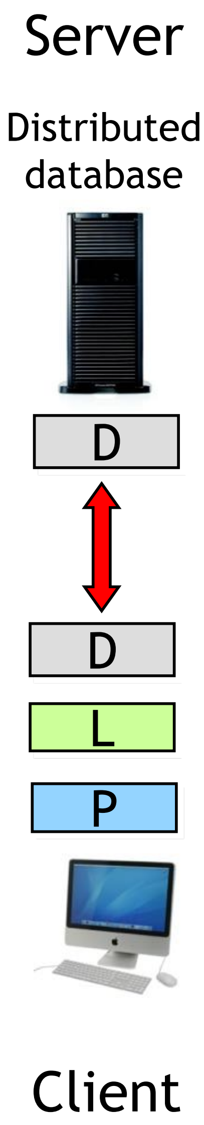
- Traditional approach for database applications
- Multiple application systems use the same database.
- Data management can also be distributed across multiple servers.
- Problem: There are several implementations of the popular database query language „SQL“ with many proprietary extensions and differences.
- Examples:
  - Customer Information System
  - Dropbox App
  - DB Navigator App (previously)



Distributed Database

Data management is distributed between server and client:

- Two incarnations of a distributed database exist:
  - Partitioning of data storage between server and client
    - Organisational structure: Centralised directory of an enterprise vs. personal address book
    - Frequency of use: Current business figures vs. archive
    - Access time: Current stock market values vs. archive
    - …
  - Partitioning of database management system (DBMS) between server and client
    - Data access functionality (frequently used) on the client
    - Database administration (less frequently used) on the server
    - Examples:
      - Here Maps App
      - Navigon App


Server

Distributed database

D

D

L

P

Client

Review of the Client/Server Architecture

- Advantages
  - Can be designed and extended flexibly
  - High interaction and communication capabilities
  - Dependability through redundant resources
- Disadvantages
  - High server workload because of multi-user access
  - High planning and coordination efforts
  - High network bandwidth required
  - High administrative workload

Cloud Computing Architecture

Internet-centric computing architecture:

- Providers are offering complex services based on hard- and software in an abstract form.
- Storage, computing power, or complex services can be accessed by client via defined interfaces via the Internet.
- Underlying hard- or software of a cloud is not relevant for a client.
- Types of cloud computing services
  - Infrastructure as a service
  - Platform as a service
  - Software as a service
- Providers, e.g.
  - Amazon, Google, Microsoft, Deutsche Telekom, etc.

Cloud Computing Architecture

- Advantages
  - Information systems become highly scalable.
  - Central data storage and backup
  - Cost efficient (one has only to pay for the actually used computing power and time)
  - Anytime, anywhere access to applications and data
  - Allows to run sophisticated applications on low-powered systems (e.g. mobile devices’ voice recognition systems)

- Disadvantages
  - Enterprises or end users have to rely on the cloud service provider and the legal and political environment.
  - Potential threats
    - Data leakage
    - Data unavailability
    - Provider bankruptcy, lock-in effects
    - Internet connection failures

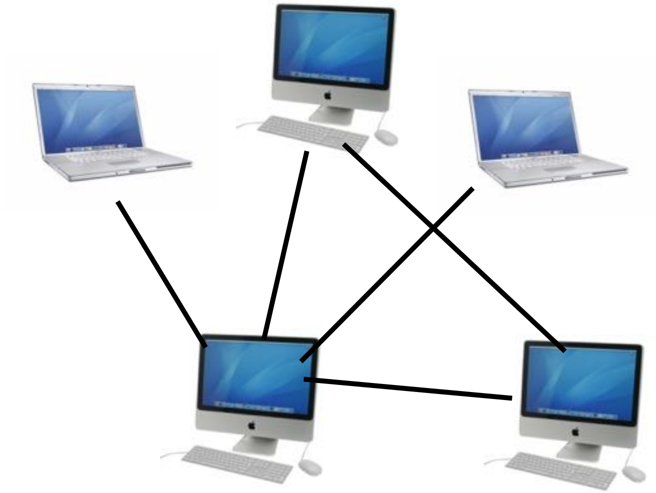
Peer-to-Peer Architecture

Network of computers with equal capabilities

- Properties
  - No central instance coordinating the required interactions
  - No centralised database
  - Peers act autonomically.
  - Each peer is only aware of those other peers it is currently communicating with.
  - Peers, connections, and information flows within this concept are not guaranteed.

- Advantage
  - Required resources are provided by many parties (e.g. for the distribution of large files)

- Disadvantages
  - High complexity
  - Requires critical mass of peers



Edge Computing Architecture

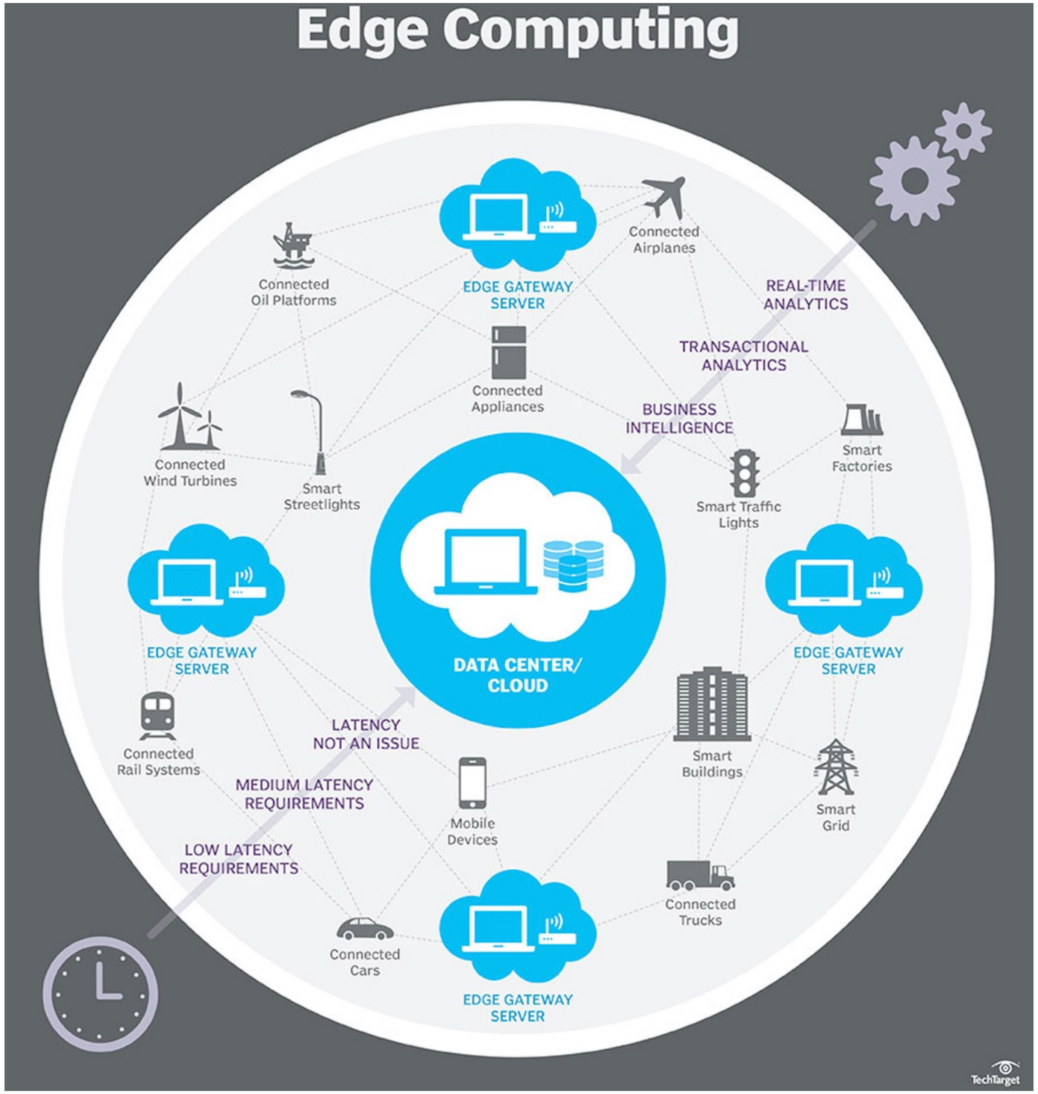
Pushing "intelligence" to the edge of the network

- Why edge computing?
  - Proliferation of IoT devices producing data to be processed
  - Centralized nature of cloud architectures imposes limitations.
  - Distance to data center impacts clouds' quality of service.
  - Steady decline in the cost of processing power & appearance of intelligent endpoint devices that sense and can make inferences

- What is edge computing?
  - Distributed approach to computing at/near network-endpoints
  - Heterogeneous nomenclature [edge (2004-), fog (2012-) and mist (2015-) computing] due to multiple interests and approaches
  - None of them synonymous with cloud computing

Edge Computing Architecture: Advantages and Applications

- Lower core network load and transmission costs since less / pre-processed data is transmitted
- Reduced load on cloud server / data centre and more efficient resource use possible
- Reduced latency key enabler of use-cases like autonomous driving
- New functionalities provided by intelligent endpoint devices



Source: Tech Target, 2017

Edge Computing Architecture: Disadvantages

- Security challenges
  - Distributed architecture increases number of attack vectors.
  - The more "intelligent" the device, the more vulnerable to infections and exploits (e.g. integrated webserver)
  - IP address spoofing, man-in-the-middle attacks
- Trust and authentication concerns
- Risk of a configuration drift when inferior device management solutions are implemented
- Fixed physical location and cost of hardware
- "Hidden" licensing costs of endpoint devices (base version vs. additional functionalities)
- Sometimes adding unnecessary overhead and complexity to the system, as not always needed for IoT applications

Literature

- Hennekeuser J.; Peter G. (2004) "Rechner-Kommunikation für Anwender", Springer Verlag, Berlin.
- Schwickert, A. (2003) "Grundzüge der Wirtschaftsinformatik", Universität Gießen.
- Tech Target (2017) "Edge Computing", http://searchdatacenter.techtarget.com/definition/edge-computing, last visited 14-09-2017
- WikiCommons (2011), http://en.wikipedia.org/wiki/Wikimedia_Commons, last visited 03-07-2013