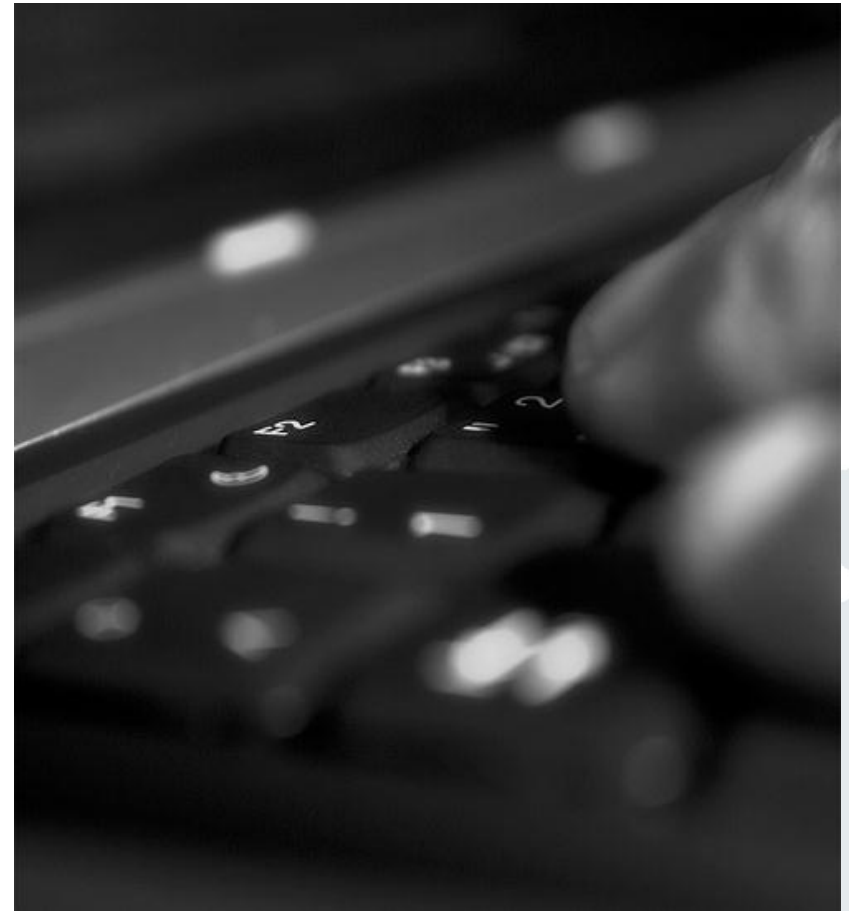


Lecture 3 Business Informatics 2 (PWIN)

Information Systems II Models and Architectures

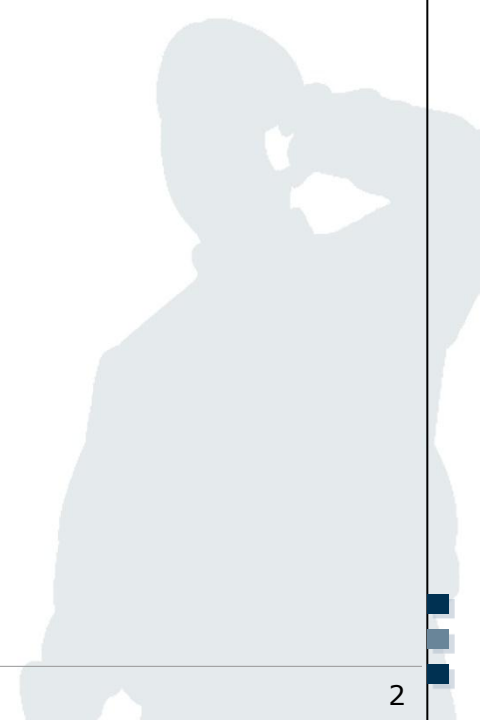
SS 2011

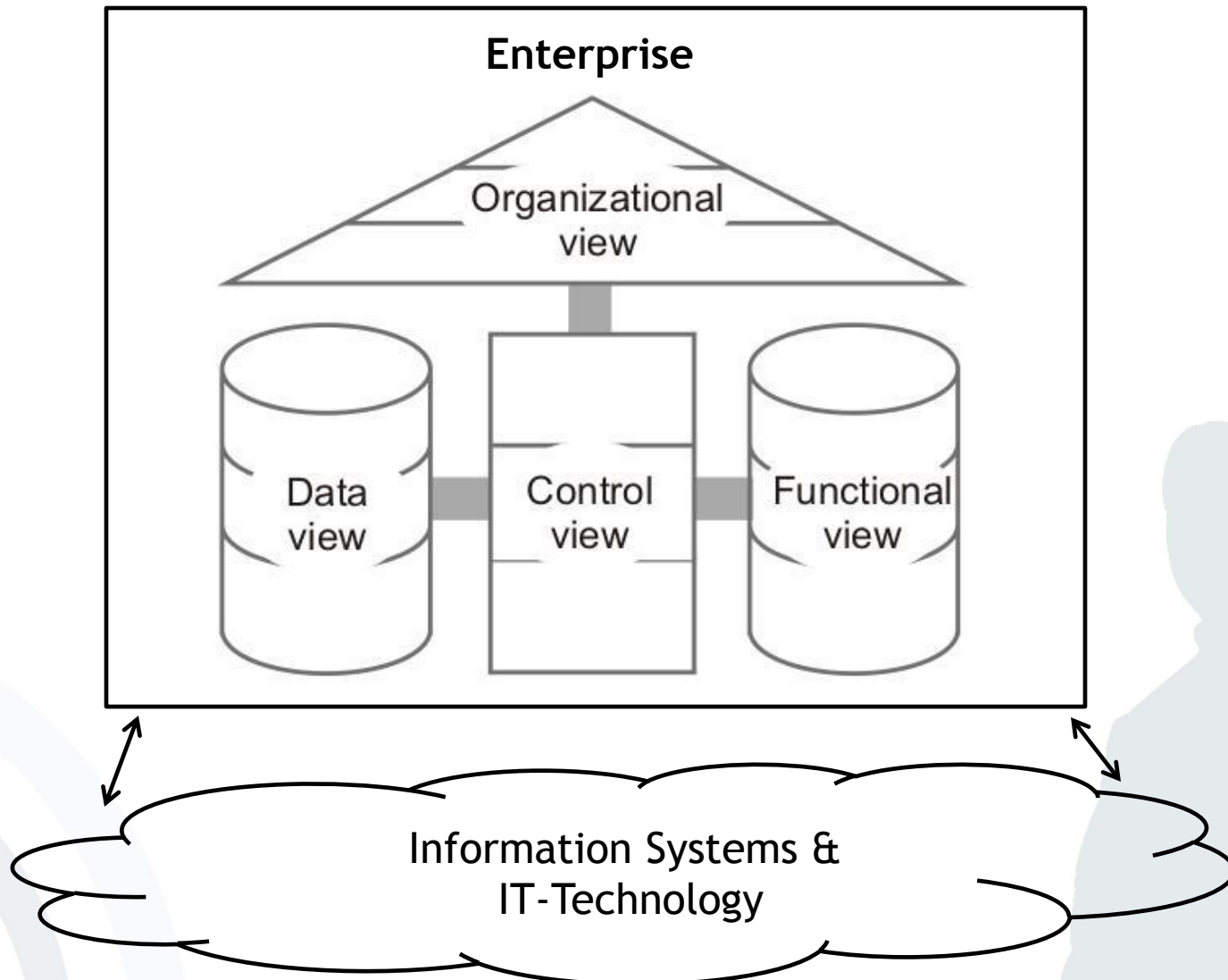
Dr. Andreas Albers
www.m-chair.net

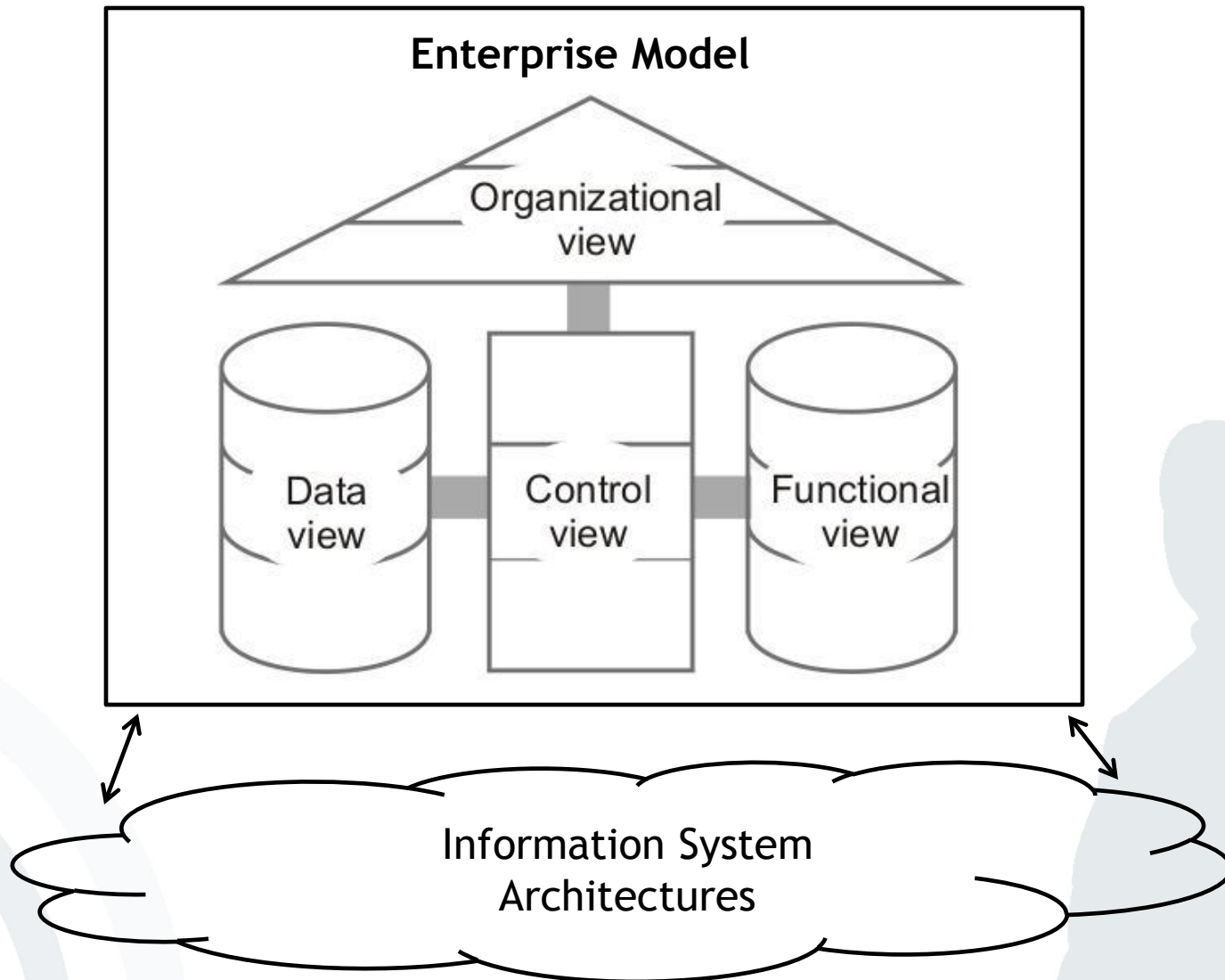


Jenser (Flickr.com)

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





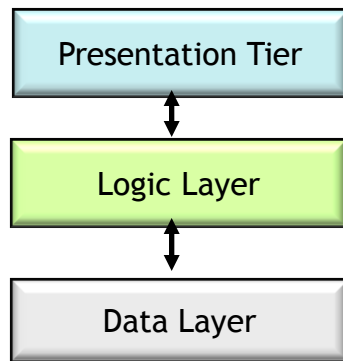


- Enterprise Models vs. IS Architecture (Models)
- Structural Models for IS Architectures
- IS Architecture Concepts

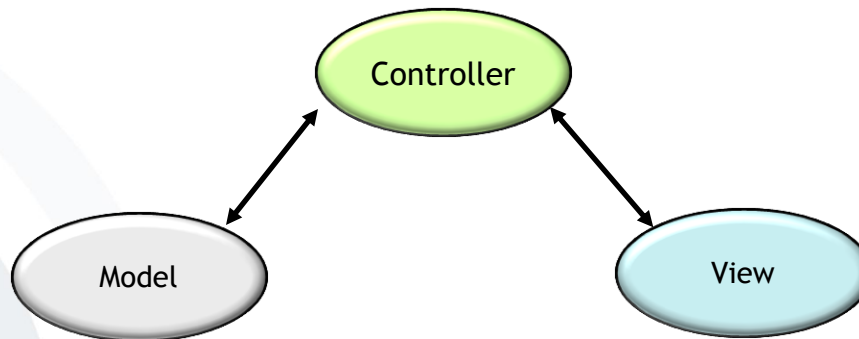
- Minimisation of Complexity for IS Components
- Scalability of IS Components
- Portability of IS Components
- Maintainability of IS Components
- Standardisation of IS Components
- Well-defined Interfaces between IS Components
- Independence of IS Components

Modularisation of IS Components

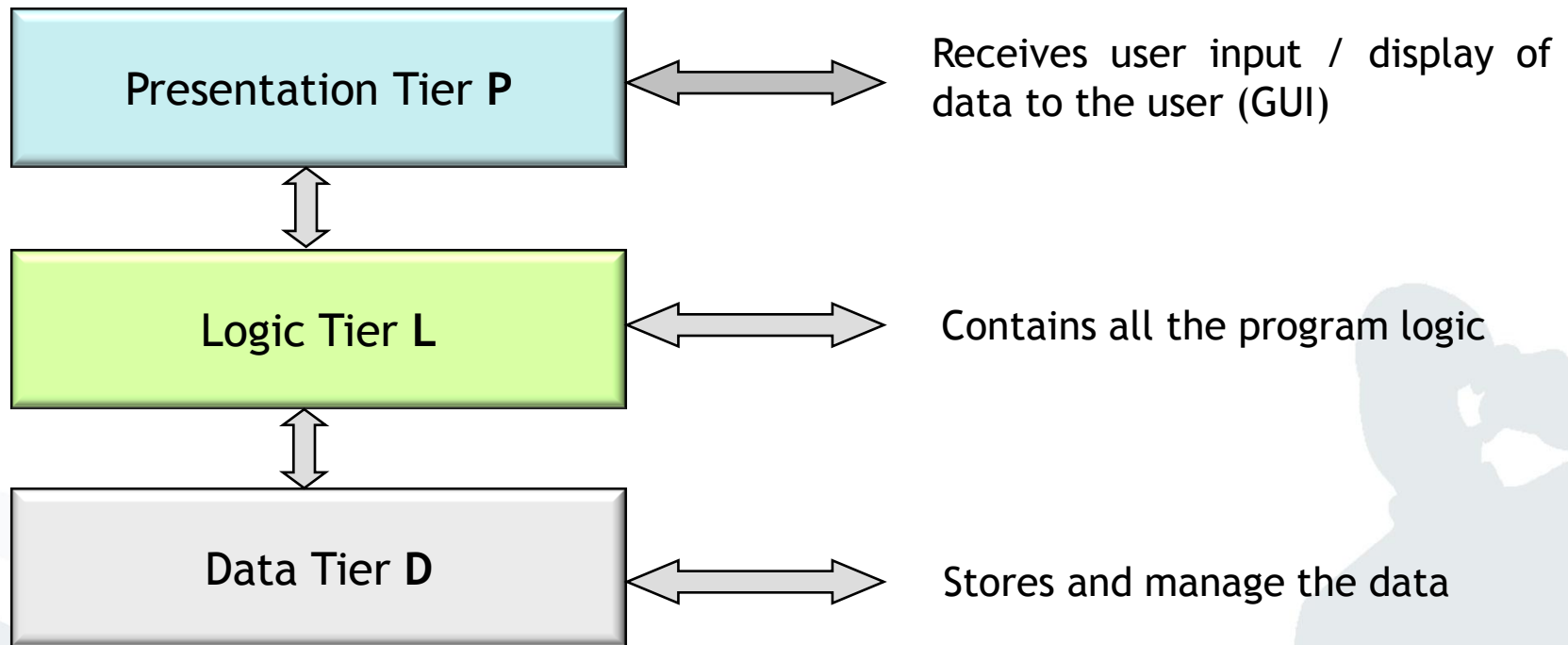
- Three-Tier Concept



- Model-View-Controller (MVC) Concept

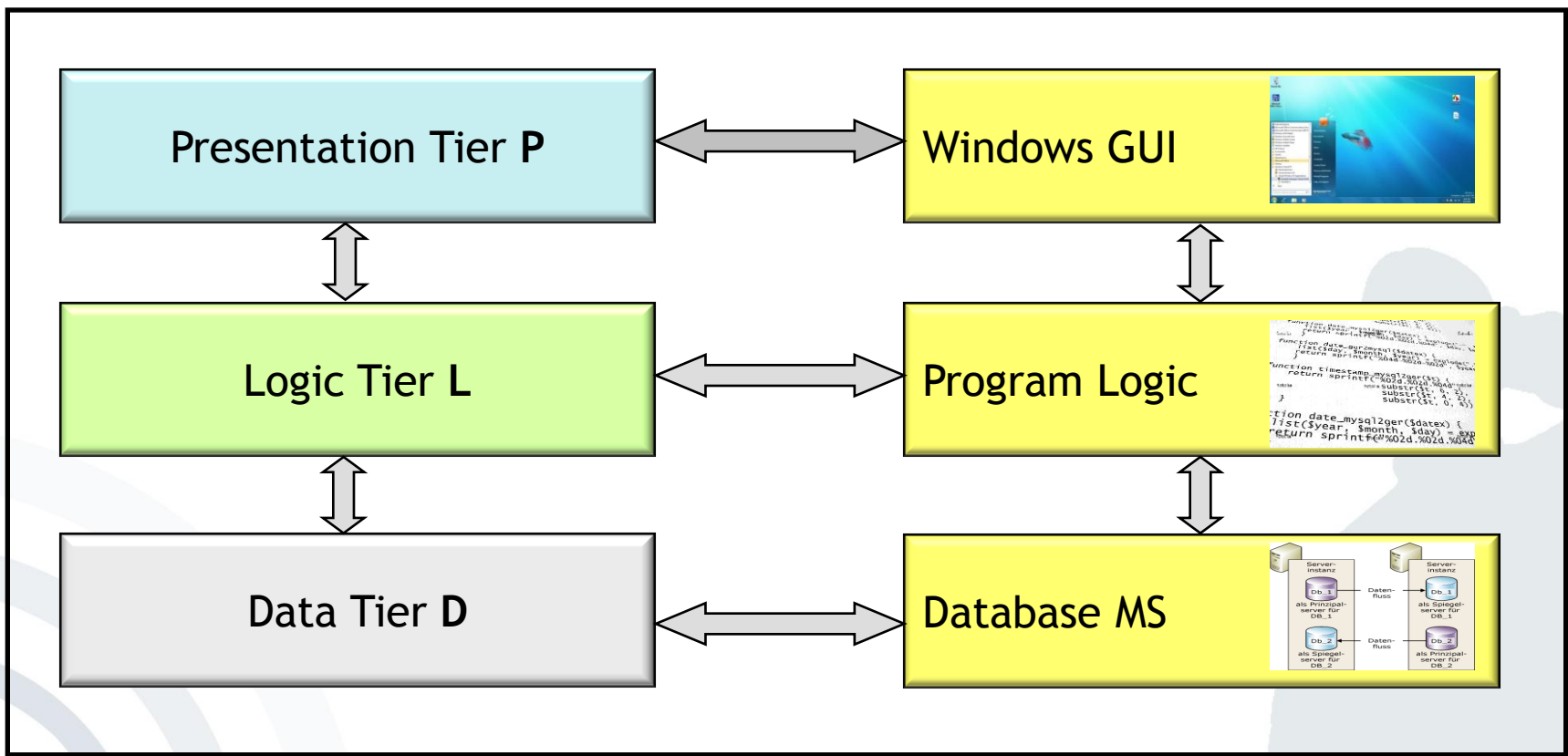


Three-Tier Concept

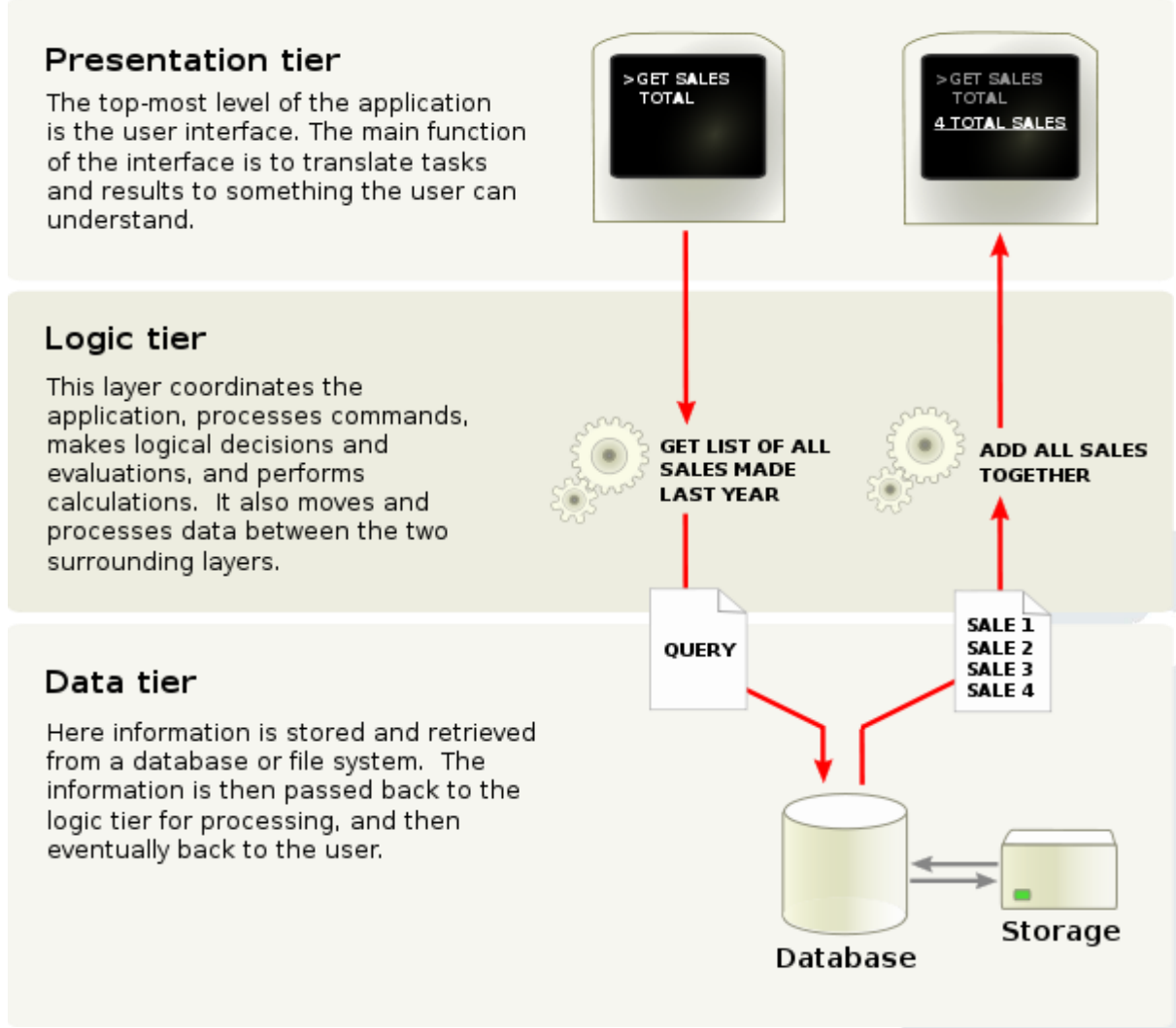
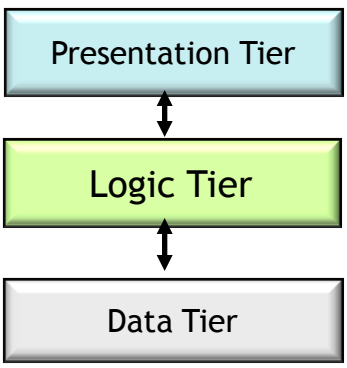


Three-Tier Concept Example (1)

Conventional IS



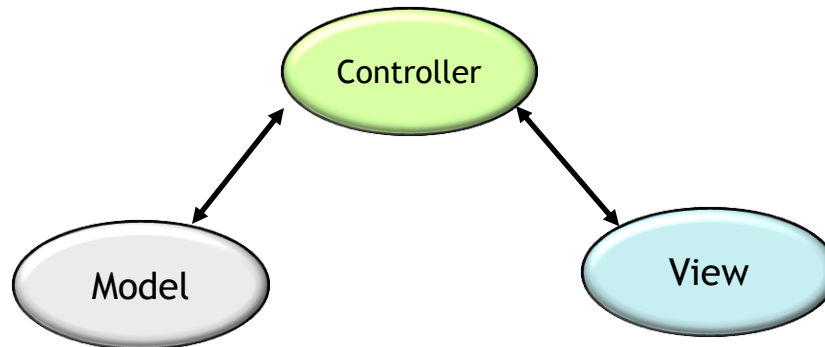
Three-Tier Concept Example (2)



Source: Wiki Commons, 2011

Model-View-Controller Concept

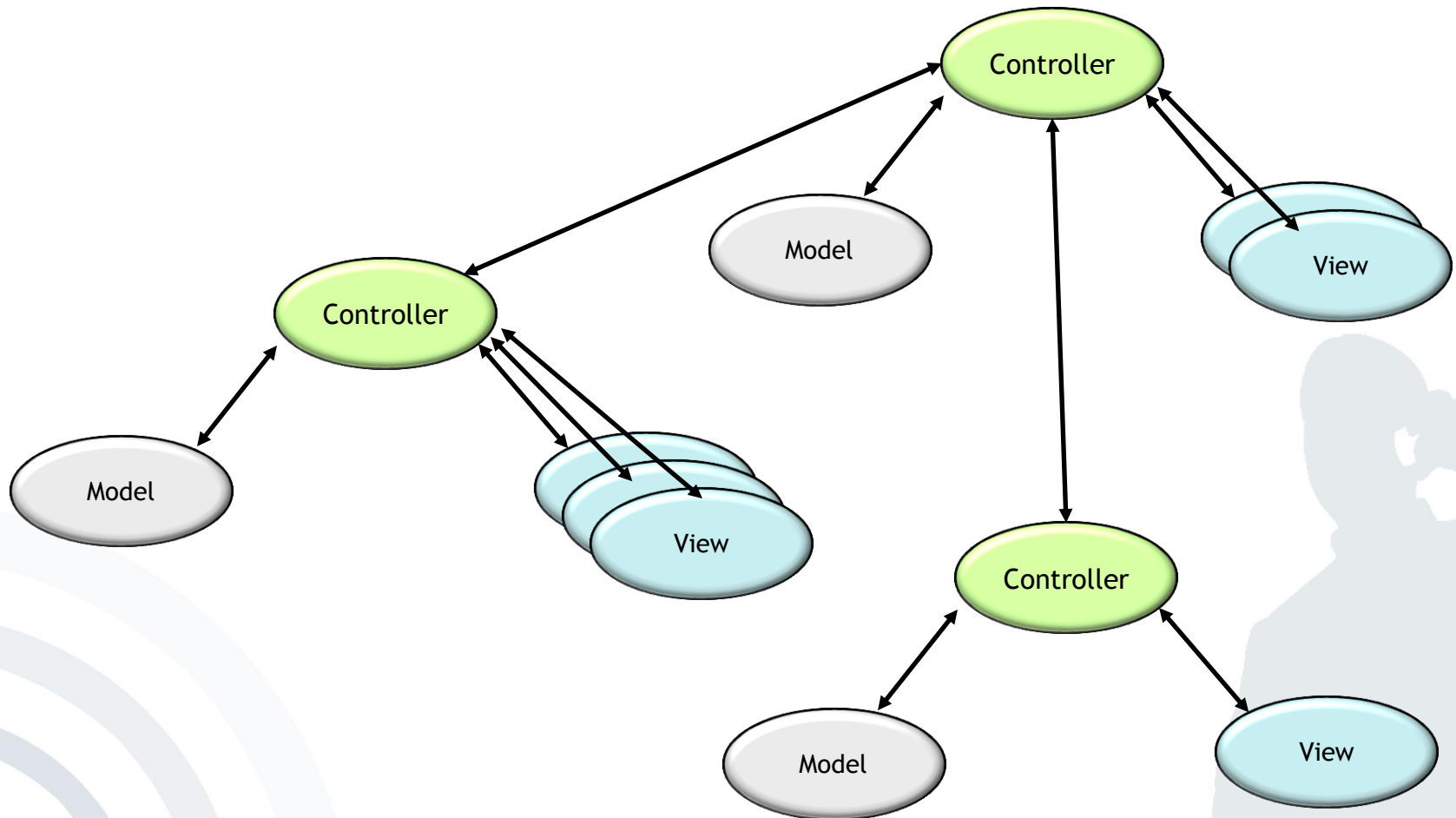
Controller controls view(s) and initiates the relevant data updates



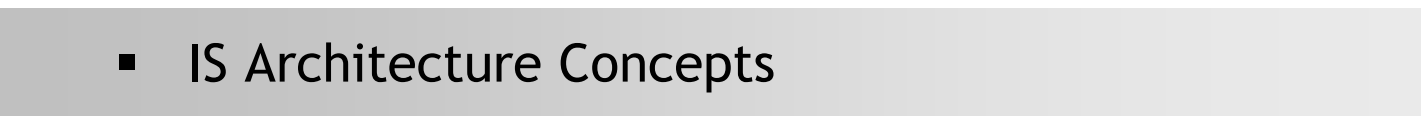
Manages data and, if applicable, contains the program logic

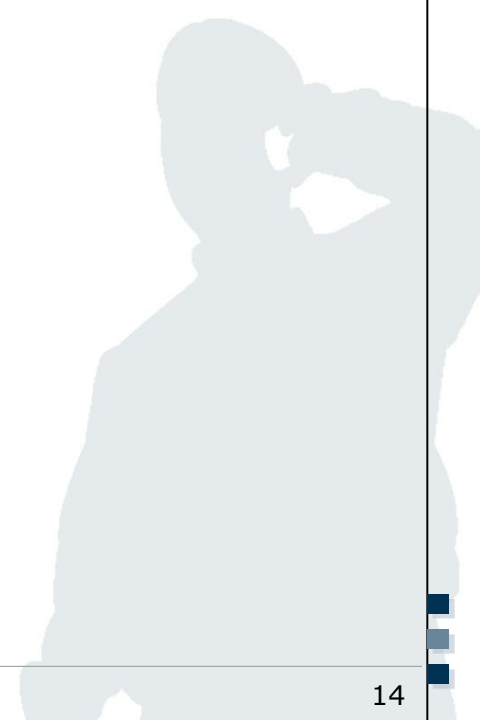
Receives user input / displays data from *model* to the user (GUI)

More Complex Model-View-Controller 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 (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.

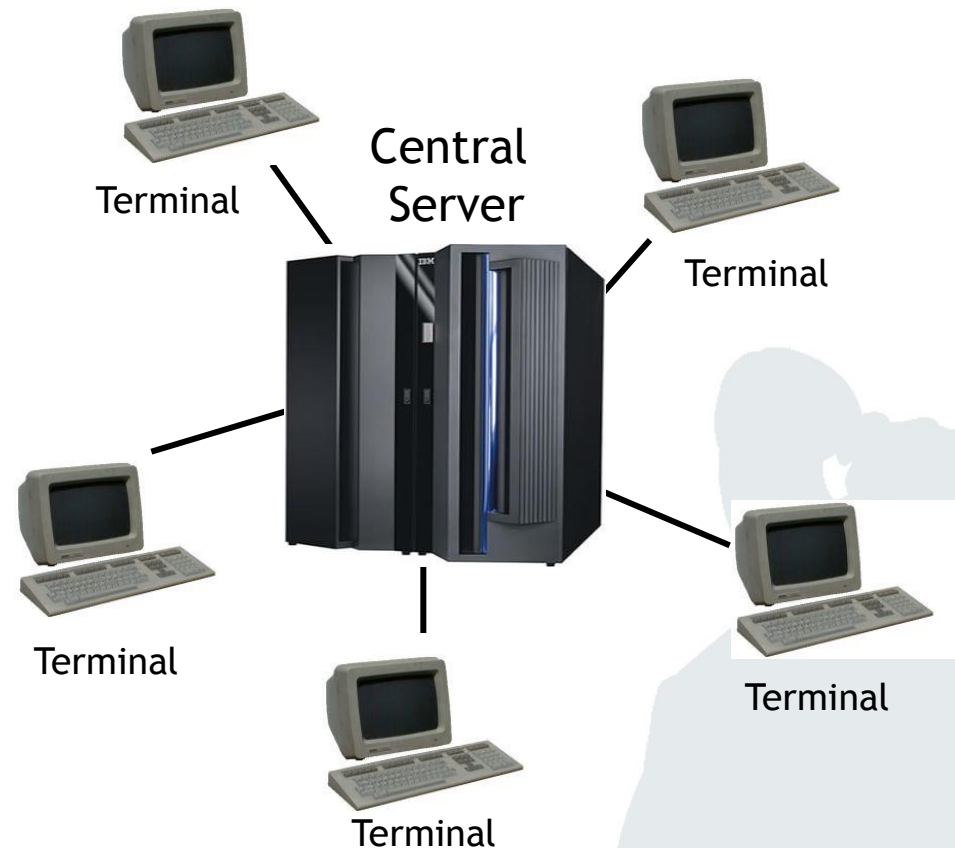
- Enterprise Models vs. IS Architecture (Models)
 - Structural Models for IS Architectures
 - IS Architecture Concepts
- 



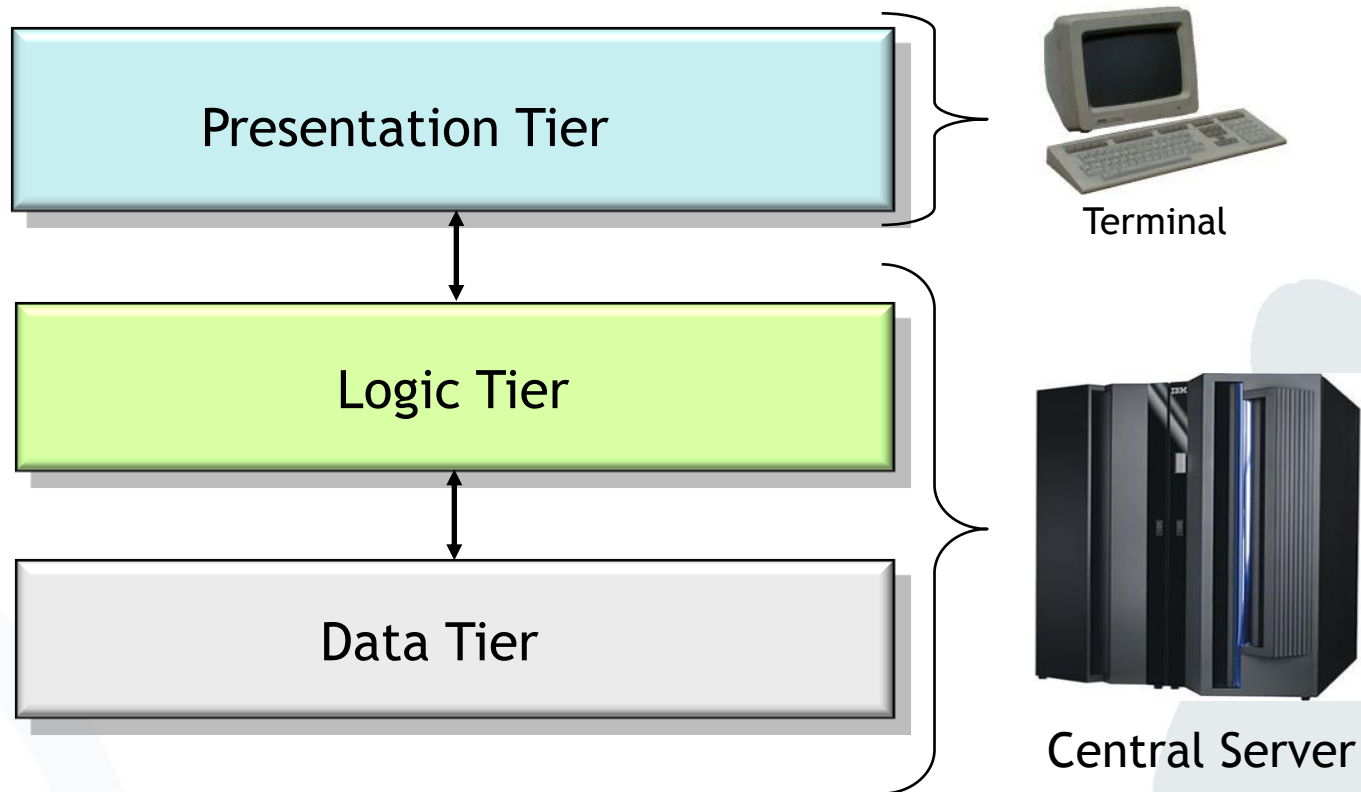
- **Central Server Architecture**
Low-feature terminals (receiver of services) attached to a powerful central computing unit (provider of services)
- **Client / Server Architecture**
Network of computers, which can take the role of a server (provider of services), a client (receiver of services) or both.
- **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)

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



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

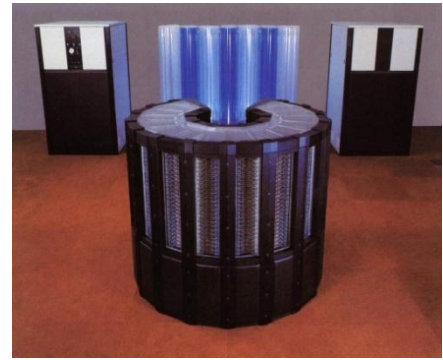
- Issues
 - Single Point of Failure
 - Fixed Network Structure
 - Monolithic
 - Cost-intensive Central Servers
 - Problematic in case of huge traffic and amounts of data

Hardware



take it to the nth

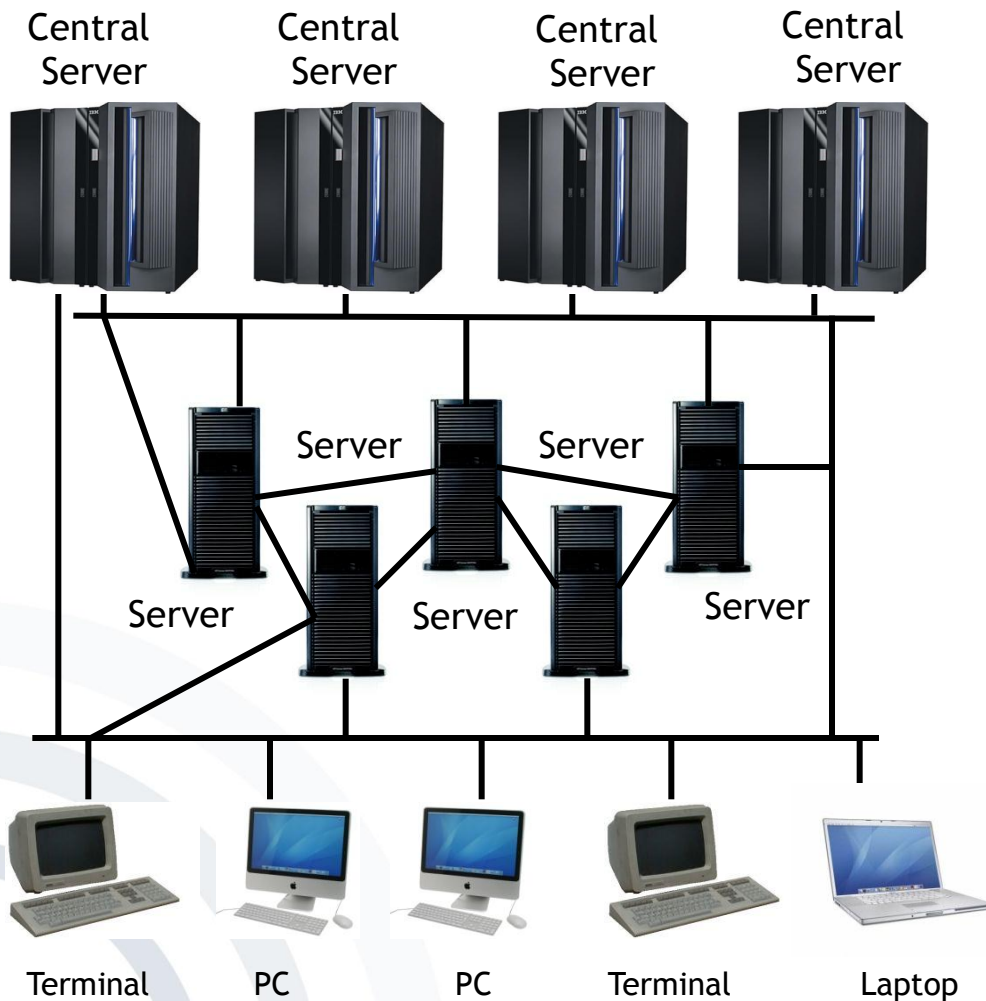
i n v e n t



Operating Systems

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

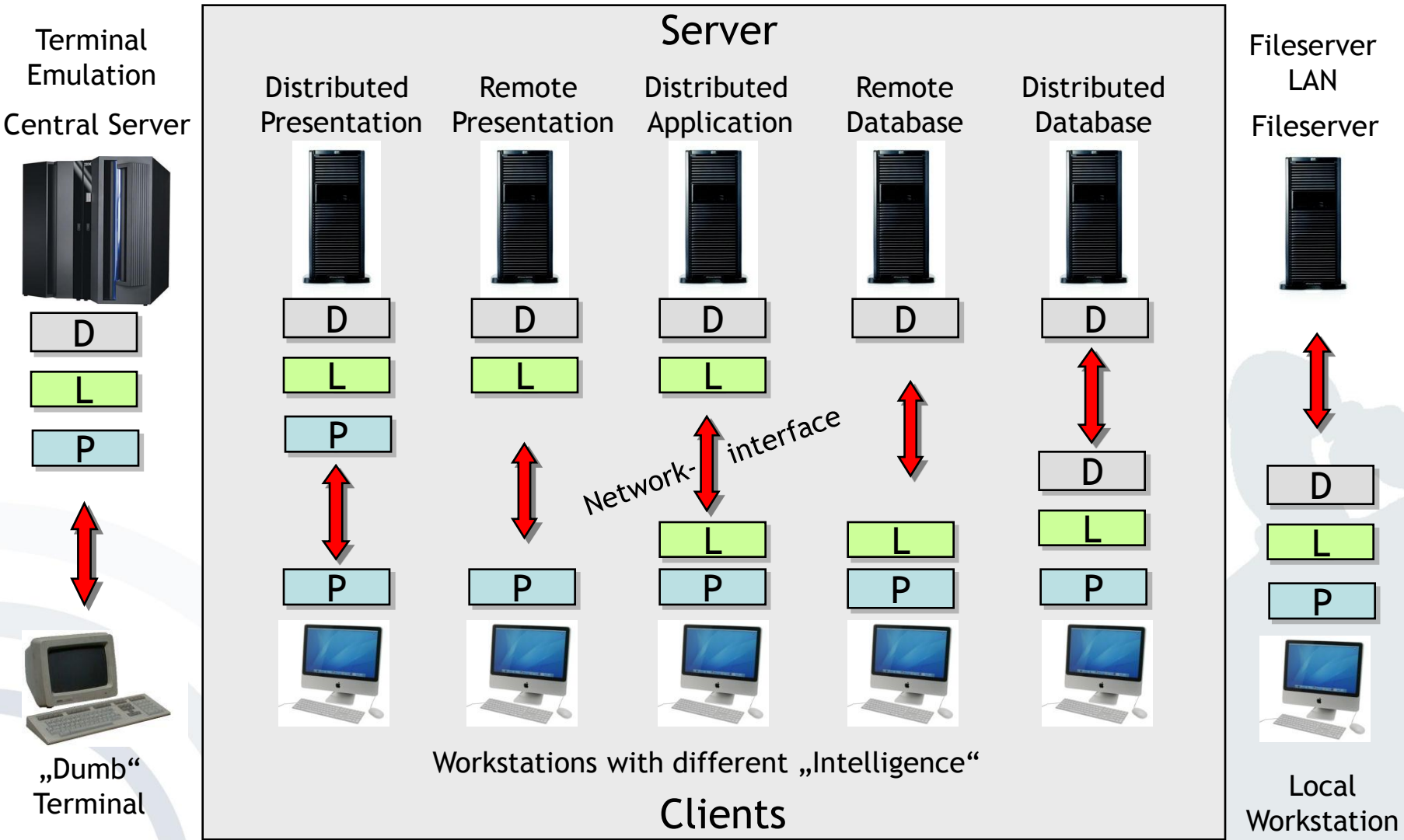
Client/Server Architecture



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



Client/Server Architecture along the Three-Tier Structural Concept

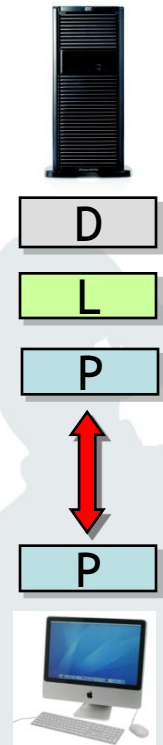


Source: Based on Hennekeuser, 2004

Division of the Communication on Server and Client

- **Abstract part of the communication (Server)**
Objects (e.g. a window) are created abstract, i.e. without any concrete representation and functionality.
- **Distinctive part of the communication (Client)**
Abstract objects are created and represented in a platform-specific manner.
- **Benefits of this approach**
Heterogeneous application systems can be integrated into a unified user interface or used on different platforms.
- **Application example: X-Windows**
A user interface using X-Windows can be represented on multiple platforms.

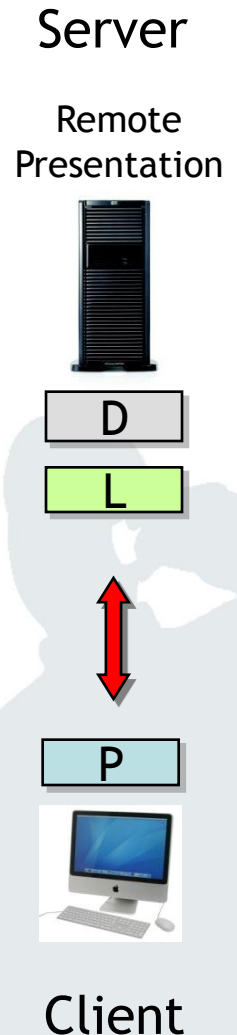
Server
Distributed
Presentation



Client

Presentation is outsourced to the Client

- Outsourcing of the communication or presentation to the client become specifically beneficial in case of a connection to a Central Server without an own user interface.
- Clients are able to run on different platforms.
- User interfaces can individually customised according to a user's needs. (e.g. GUI)
- Client can not be a „dumb“ terminal.
- Examples: Citrix XenDesktop, TeamViewer



Application functions are partitioned:
Central vs. decentralized

- Central application functions are hosted on the server in order to be available for everyone.
- Decentralized applications reside individually on the client.
- Central application functions will only be used on demand.
- Advantages: Development and maintenance of application functions get simplified; complexity is reduced
- Example: Groupware

Server

Distributed
Application



D

L



L

P



Client

Data management resides on the Server

- Traditional partition for database applications.
- Multiple application systems use the same database.
- Data management can also be distributed across multiple servers.
- Problem: DB-Query-Standard „SQL“ contains proprietary extensions and differences.
- Classic example: Customer Information System

Server

Remote
Database



D



L

P



Client

Data management is distributed between Client and Server

- Two incarnations of a distributed database exist
 - Partitioning of data into central (Server) and decentralize (Client) stored data
 - Organisational structure: Centralized address book of an enterprise vs. personal address book
 - Frequency of Use: Current business figures vs. business archive
 - Access Times: Current stock market values
 - ...
 - Partitioning of the data management (DBMS) between Client und Server
 - Data access functionality (frequently used) on the Client
 - Database administration (less frequently used) on the Server

Server

Distributed Database



D



D

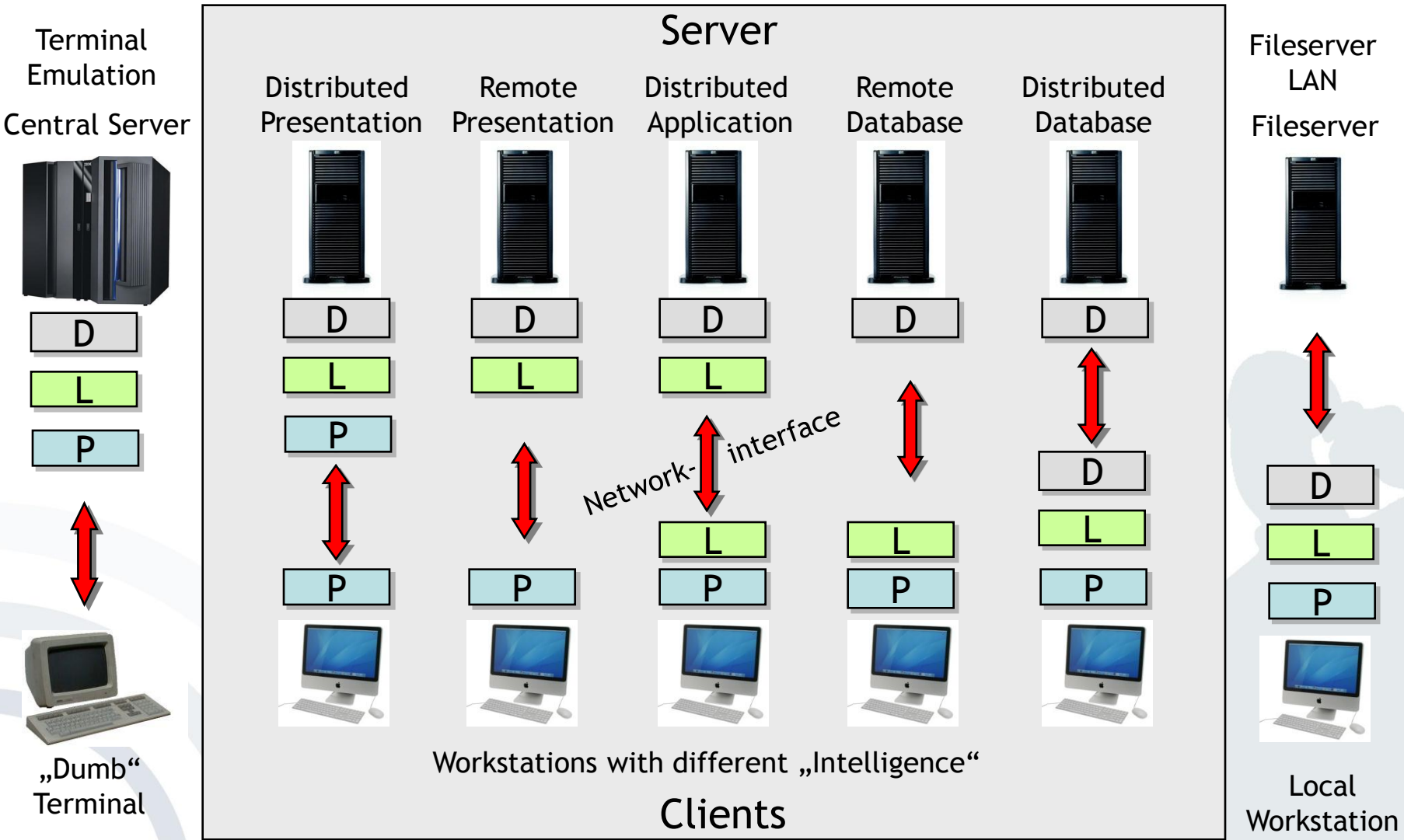
L

P



Client

Client/Server Architecture along the Three-Tier Structural Concept



Source: Based on Hennekeuser, 2004

- Benefits
 - Flexible designable and extendable
 - High interaction and communication capabilities
 - Fail-safety through redundancy
- Issues
 - High workload because of multi-user access
 - High planning and coordination efforts required
 - High network bandwidth required
 - High administrative workload

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 Cloud Computing Services
 - Infrastructure as a Service
 - Platform as Service
 - Software as Service
- Providers, e.g.
 - Amazon, Google, Microsoft, etc.

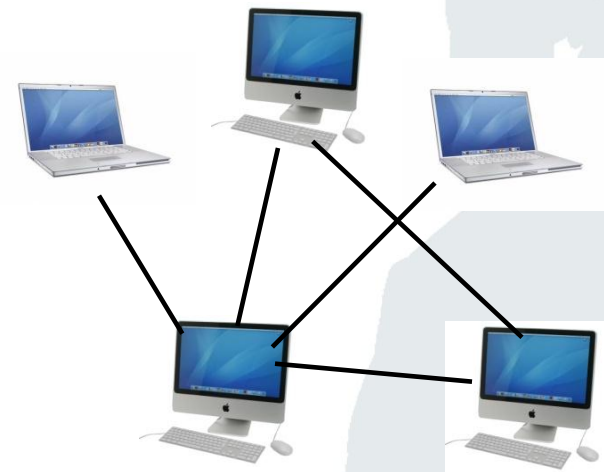


- Advantages
 - Information system become highly scalable
 - Central data storage and backup
 - Cost efficient (one has only to pay for the actually used computing power and time)
 - Anytime, anywhere access to applications and data
 - Allows to run sophisticated applications on low-powered systems (e.g.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

- Disadvantages
 - Enterprises or end users have to rely on the Cloud Service Provider
 - Potential Threats
 - Data Security, Data Privacy
 - Provider Bankruptcy, Lock-in Effects
 - Internet Connection failures

Network of computers with equal rights

- Properties
 - No central instance coordinating the required interactions
 - No centralized database
 - Peers act autonomic.
 - Each peer is only aware of those other peers to which he is currently communicating with.
 - Peers, connections and information within this concept are not guaranteed for the participants.
- Advantage
 - Required resources are provided by many parties (e.g. for the distribution of large files)
- Disadvantage
 - High complexity of Peer-to-Peer systems
 - Requires critical mass of peers
 - Security issues





- Hennekeuser J.; Peter G. (2004) "Rechner-Kommunikation für Anwender", Springer Verlag, Berlin.
- Schwickert, A. (2003) "Grundzüge der Wirtschaftsinformatik", Universität Gießen.
- WikiCommons (2011), http://en.wikipedia.org/wiki/Wikimedia_Commons, last visited 01-04-2011

