

# DEVELOPMENT OF INFORMATION SYSTEMS


## Lecture 2

# Repetition of the last lecture

## Information system is:


The interaction between people, processes and data.

An information system is designed to process (retrieve, transmit, store, search, manipulate, display) information.

Several thin, parallel white lines are drawn diagonally across the bottom right corner of the slide, extending from the middle of the right edge towards the bottom left.

# Repetition of the last lecture

## When did it start (and when will it end)

- Wages (payroll system), Warehouses, ...
    - High human labour costs
    - Error rate
  - Auction servers, online stores, ...
    - Efficiency, speed, distance, ...
  - It never ends?
- 
- A series of three parallel white diagonal lines in the bottom right corner of the slide, extending from the middle of the right edge towards the bottom left.

# Repetition of the last lecture

## Classification of IS

Different types of agendas:

- Economic
- Personal
- Warehouse
- Documentographic (library, records management)
- Geographics (GIS)
- School, students
- ERP, HRM, CRM, Project management, Supply chain management

# Formalised vs computerised IS

## Formalised

- Not formalisation in the sense of mathematics...
- It is possible to work with information on an informal basis (especially lacking predefined and binding form of working with information).

## Computerised (computer based)

- Information systems provide support to increase efficiency of working with information (is not always true ☹).

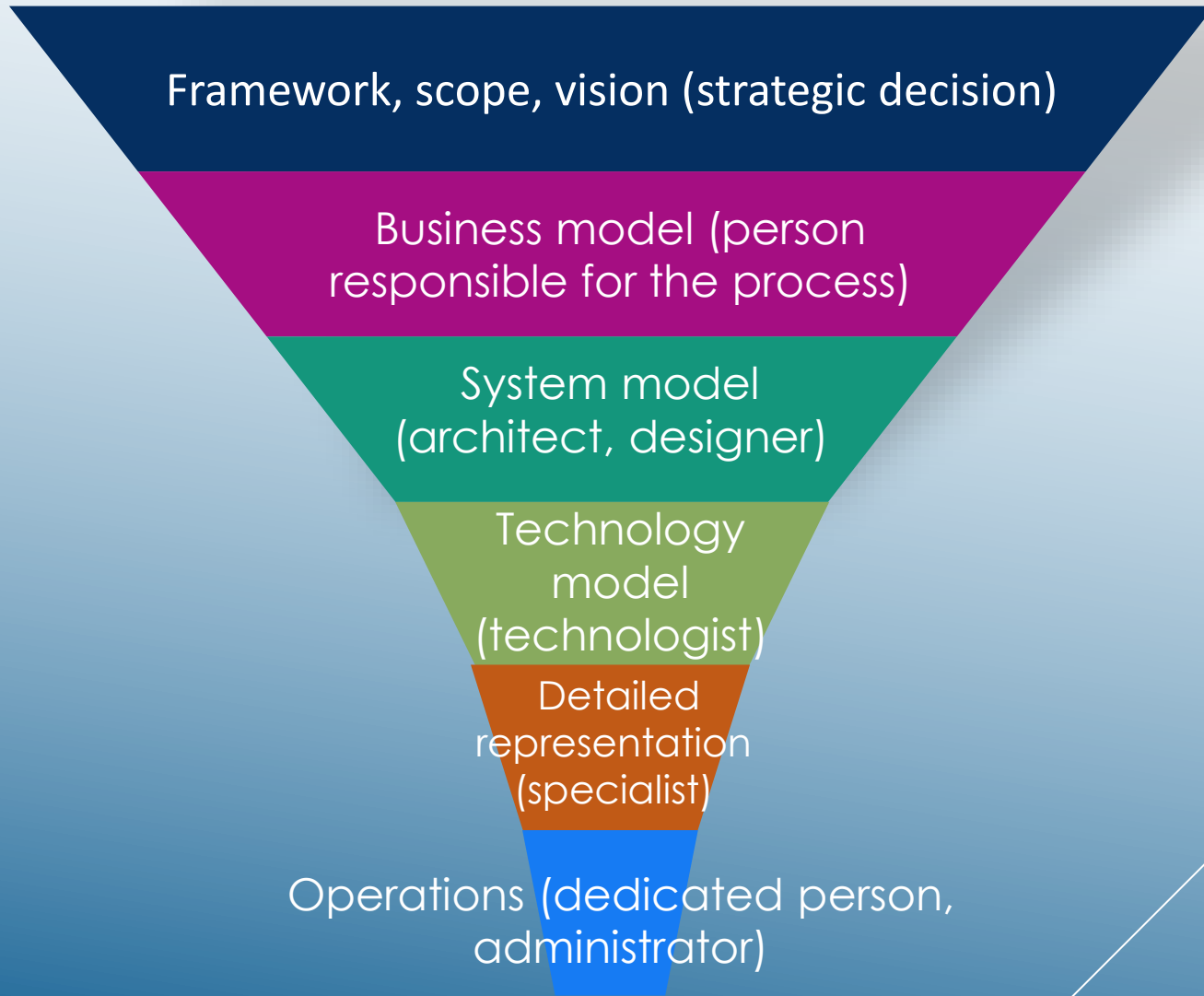
# Domain of Information System

- By the **domain** we understand a group of related "things, concepts and terms" in the sense of a customer or user point of view.
- The domain includes data, processes, users, etc. that are characterized by specific concepts.
- By using these terms, we determine which domain we are in (e.g. in the bank domain we use terms such as: account, transaction, money, cashier, credit card, etc.).

# Key questions related to the domain

- **WHAT?** It is information in the sense of data.
- **HOW?** It is about the processes performed with the information.
- **WHERE?** The places where information is worked with.
- **WHO?** Who works with information and in what role.
- **WHEN?** When and on what impulses information is worked with
- **WHY?** It is about aims and rules to achieve the requested goals.

# From the general to the specific





# The Zachman Framework for Enterprise Architecture™

## The Enterprise Ontology™


Version 3.0



# Simpler version of the Enterprise Ontology

	DATA <i>What</i>	FUNCTION <i>How</i>	NETWORK <i>Where</i>	PEOPLE <i>Who</i>	TIME <i>When</i>	MOTIVATION <i>Why</i>
Objective/Scope (contextual) <i>Role: Planner</i>	List of things important in the business	List of Business Processes	List of Business Locations	List of important Organizations	List of Events	List of Business Goal & Strategies
Enterprise Model (conceptual) <i>Role: Owner</i>	Conceptual Data/ Object Model	Business Process Model	Business Logistics System	Work Flow Model	Master Schedule	Business Plan
System Model (logical) <i>Role: Designer</i>	Logical Data Model	System Architecture Model	Distributed Systems Architecture	Human Interface Architecture	Processing Structure	Business Rule Model
Technology Model (physical) <i>Role: Builder</i>	Physical Data/Class Model	Technology Design Model	Technology Architecture	Presentation Architecture	Control Structure	Rule Design
Detailed Representation (out of context) <i>Role: Programmer</i>	Data Definition	Program	Network Architecture	Security Architecture	Timing Definition	Rule Speculation
Functioning Enterprise <i>Role: User</i>	Usable Data	Working Function	Usable Network	Functioning Organization	Implemented Schedule	Working Strategy

# IS development main steps

- **Vision**
  - **Analysis** (user requirements)
  - **Logical design** (functional req)
  - **Technological design** (non-functional req)
  - **Development**
  - **Deployment and operation**
- 
- Several thin, white, parallel diagonal lines are located in the bottom right corner of the slide, extending from the bottom edge towards the right edge.



# IS development life cycle

## Vision document

The Vision is created early in the inception phase.

Provides a high-level executive summary for a project or other undertaking.

Captures the essence of the envisioned solution in the form of high-level requirements and design constraints that provide stakeholders an overview of the system to be developed from a behavioral requirements perspective.


Contain an outline of the envisioned core requirements, it provides the contractual basis for the more detailed technical requirements. It is much shorter and more general than a product requirements document or a marketing requirements document, which outline the specific product plan and marketing plan respectively.

# IS development life cycle

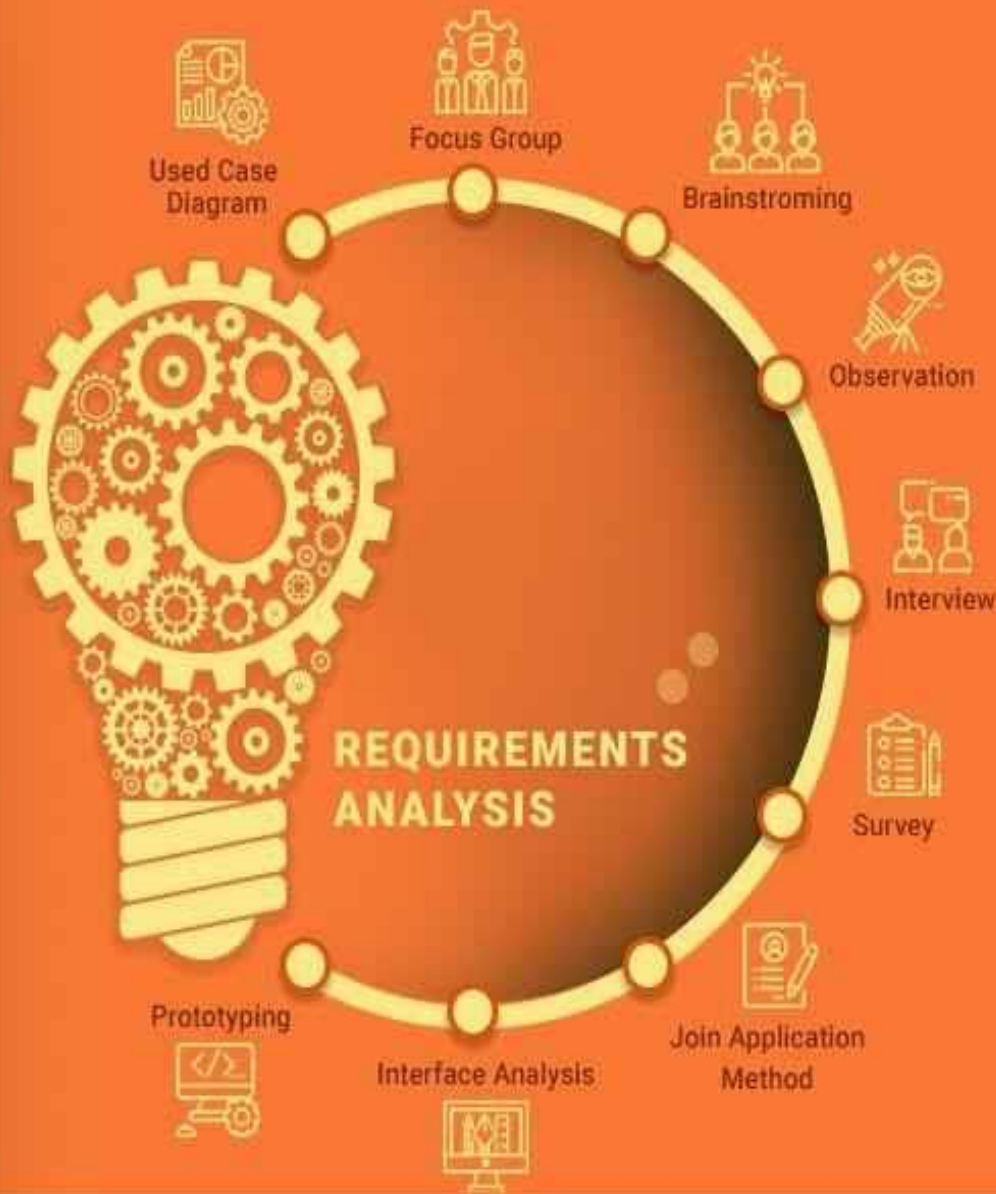
## Analysis (requirements)

Is the process of defining the expectations of the users for an application that is to be built or modified. It involves all the tasks that are conducted to identify the needs of different stakeholders.

Therefore requirements analysis means to analyze, document, validate and manage software or system requirements..

- Eliciting requirements
  - Analyzing requirements
  - Requirements modeling
  - Review and retrospective
- 

# IS development life cycle



## Techniques

- BPMN
- UML
- Flowchart
- Data flow
- Gap analysis

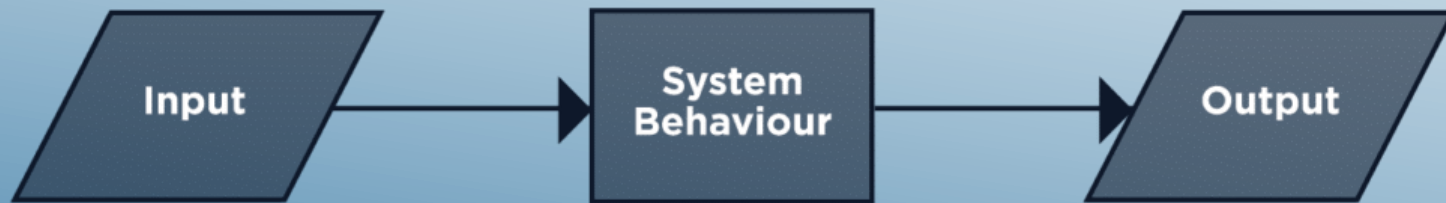
## REQUIREMENTS ANALYSIS

Understanding the basics- Definition, Process, & Requirement Analysis Techniques

# IS development life cycle

## Logical design - functional requirements


Functional requirements define the basic system behaviour. Essentially, they are **what** the system does or must not do, and can be thought of in terms of how the system responds to inputs. Functional requirements usually define if/then behaviours and include calculations, data input, and business processes.



Functional requirements are features that allow the system to function as it was intended. Put another way, if the functional requirements are not met, the system **will not work**. Functional requirements are product **features** and focus on user **requirements**.

# IS development life cycle

## Logical design – tools and methods

- BPMN (business process model notation)
  - UML
    - Use-case diagrams
    - Use-case description
    - State diagram
    - Activity diagram
    - Sequence diagram
  - Brain storming
- 
- A decorative graphic consisting of several parallel white diagonal lines of varying lengths, located in the bottom right corner of the slide.



## ► How are functional requirements gathered?

Analyse Vision, brainstorming with stakeholder and end users.

- **Business requirements.** This usually contains the ultimate goal, such as an order system, an online catalogue, or a physical product. It can also include things like approval workflows and authorization levels.
- **Administrative functions.** These are the routine things the system will do, such as reporting.
- **User requirements.** These are what the user of the system can do, such as place an order or browse the online catalogue.
- **System requirements.** These are things like software and hardware specifications, system responses, or system actions.

# IS development life cycle

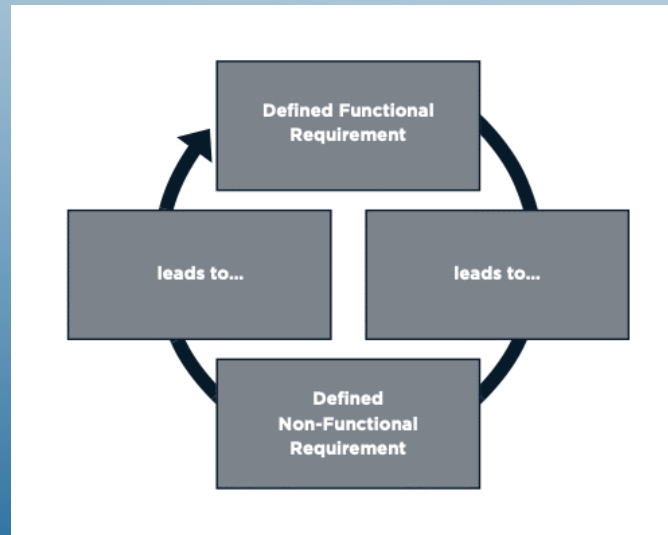
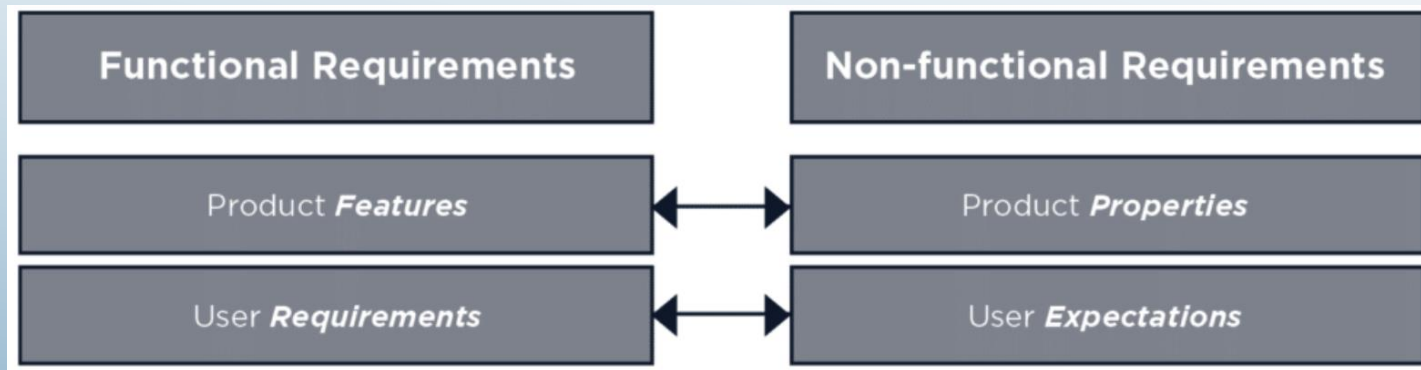
## Technological design (non-functional requirements)

While *functional* requirements define what the system does or must not do, ***non-functional*** requirements specify **how** the system should do it. Non-functional requirements do not affect the basic functionality of the system (hence the name, *non-functional* requirements). Even if the non-functional requirements are not met, the system will still perform its basic purpose.

Non-functional requirements define system behaviour, features, and general characteristics that affect the user experience. Non-functional requirements are product **properties** and focus on user **expectations**.

# IS development life cycle

## Technological design (non-functional requirements)



## ► How are non-functional requirements gathered?

After defining the functional requirements, think about the non-functional requirements, such as:

- **Usability.** This focuses on the appearance of the user interface and how people interact with it. What colour are the screens? How big are the buttons?
- **Reliability / Availability.** What are the uptime requirements? Does it need to function 24/7/365?
- **Scalability.** As needs grow, can the system handle it? For physical installations, this includes spare hardware or space to install it in the future.
- **Performance.** How fast does it need to operate?
- **Supportability.** Is support provided in-house or is remote accessibility for external resources required?
- **Security.** What are the security requirements, both for the physical installation and from a cyber perspective?

## ► How are functional and non-functional requirements written

- The most common way to write functional and non-functional requirements is a requirements specification document. This is simply a written description of the functionality that is required.
- Closely related to a requirements specification document is a work breakdown structure. This breaks down the entire process into its components by “decomposing” the requirements into their individual elements until they cannot be broken down any further.
- Another approach is user stories. These are a description of the functionality from the perspective of the end-user, and describes exactly what they want the system to do.
- Use cases are similar to user stories in that no technical knowledge is necessary. Use cases simply describe in detail what a user is doing as they execute a task.

# Examples

## Functional requirements

- When the user enters the information, the system shall send an approval request.
- The server shall log all changes to existing data.
- When 24 hours have passed since the last database backup, the server shall automatically back up the database.
- The local terminal shall automatically print a list of orders every 4 hours.
- When the local time is 08:00, the server shall email a report of open issues to all managers.
- The system shall only allow managers to view customer banking data.

# Examples

## Non-Functional requirements

- Database security shall meet HIPAA requirements.
- The layout shall allow users to reach their profile data from any page within 3 clicks.
- If a user has not changed their password for 56 days, then the system shall require a password change upon login.
- The system must accommodate a minimum of 3 million concurrent users.
- All web pages shall load within 4 seconds.
- The server room shall accommodate a future doubling of installed hardware.
- The server room shall be accessible by authorized employees 24 hours per day.
- The background colour for all screens shall be #fff4b6.
- System programming shall not use deprecated code.



# Relation Functional vs Non-Functional req

A functional requirement frequently has a related non-functional requirement. Remember that the difference is essentially the what and the how, which are both necessary.

## Functional Requirements

When a site visitor creates an account, the server shall send a welcome email.

When order status changes to fulfillment, the local printer shall print a packing slip.

The system must allow the user to fill out and submit a service form.

## Related Non-Functional Requirements

When sending welcome emails, the server must send them within 10 minutes of registration.

When packing slips are printed, they must be on both sides of 5" x 8" sheets of white paper.

When the form is requested from the server, it must load within 1 second. When the submit button is pressed, it must complete upload within 2 seconds.



# Software architecture

- „The basic organization of a software system, including its components, their interrelationships and relationships with the system environment, the principles of the design of such a system and its development.“
  - **ISO/IEC/IEEE 42010:2011 "Systems and software engineering - Architecture description „**

The architecture of an information system lies at a higher level of abstraction to include:

- The application domain view (i.e., the "customer view")
- The developer's view of the global structure of the system and the behaviour of its parts, their interconnection and synchronization
- the view of data access and data flows in the system
- the physical layout of components, etc.

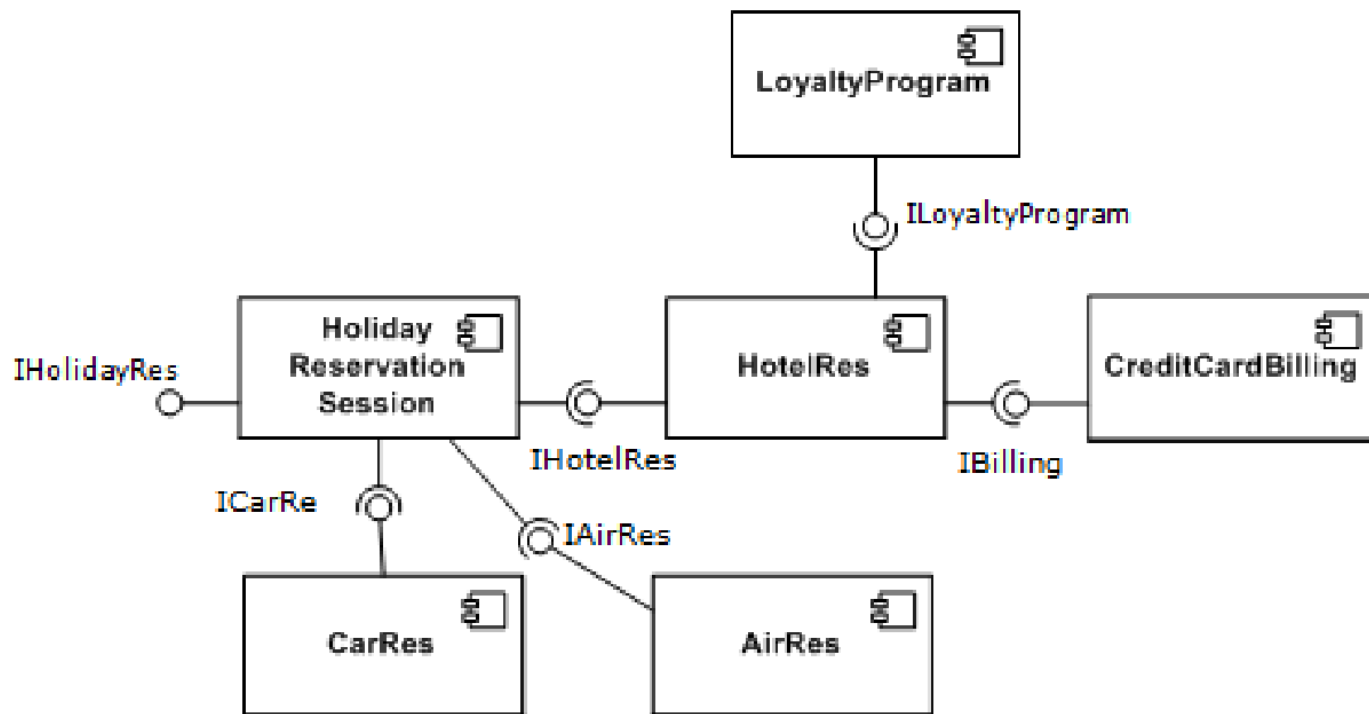
# Rules and principles

Software architecture is primarily the structure of a software system and the rules of its development.

- **Static architecture** - allows to capture only the fixed structure of the system. The structure of the system is given at design time and unchangeable during running of the system.
- **Dynamic architecture** - in comparison to static architecture, it also supports creation, modification and termination components and links at runtime according to rules determined at design time. The structure of the system changes dynamically.
- **Mobile architecture** - extends the dynamic architecture to include mobile elements, where components and links are moved at runtime according to the state of the computation

# Software component

A software component is a software package, service or generally a module that provides a particular functionality, and therefore encapsulating functionality and data.



# Architecture versus Design

**ARCHITECTURE** deals primarily with technical (other than functional) and partly functional requirements, whereas **DESIGN** is based on purely functional requirements.

The process of defining **ARCHITECTURE** uses experience, heuristics and sequential refinements and improvements. It requires a high level of abstraction for the **DRAFT** division of the system logic into separately functioning parts (with precisely defined competencies).

It cannot be learned from books. It is necessary to learn by applying procedures and continuously improve them to achieve the desired goal.

# How to proceed with IS design?

- **Decomposition**

- Identify system requirements
- Decomposition of the system into components
- Assigning requirements to individual components
- Verify that all requirements have been allocated
  - Standard Systems and software engineering - Software life cycle processes  
ISO/IEC 12207:2008

# Design versus Deployment

**DESIGN** describes the system divided into logical parts, i.e. **HOW** it works and **WHAT** it works with (classes, tables, components, services and relationships between them).

**DEPLOYMENT** describes **WHERE** the system runs (on what HW, SW platform,...)



# The three core competencies of IS design


- Communication with the user (presenting information, conveying requirements)
- Information processing and (temporary) storage.
- Permanent storage of information (data).



**N-Layered architecture**

# Exercises II.

## Discussion and acceptance on the Vision of IS


- The goal is expressed by the name (describing the IS domain).
  - Answers to 6 key questions.
  - Minimal, but complete, scope.
  - Text understandable by both parties (customer and developer).
- 
- Several thin, white, parallel diagonal lines are drawn in the bottom right corner of the slide, extending from the middle of the right edge towards the bottom left.



# Excercise II

List of functional requirements as a set of use-case - discussion.

Design two classes with association and prepare simple implementation from interface to stored data.

Several thin, white, parallel diagonal lines are located in the bottom right corner of the slide, extending from the right edge towards the bottom.

# Lecture checking questions 1/2

- Describe what is meant by the term information system, what it solves. Give examples.
- Describe what is meant by an information system domain and give examples.
- What questions must we answer when implementing an information system? Give examples of answers to each question in the context of a specific information system.
- From what perspectives (level of abstraction) can we look at an information system? And in what roles?
- What is meant by the architecture of an information system and what does it include?
- What are the differences between static, dynamic and mobile information system architecture? Give examples.

# Lecture checking questions 2/2

- What does the structure of each information system contain? Give examples.
- Describe what is meant by an information system component. Give examples.
- What is the difference between information system architecture and information system design?
- What is the correct procedure for determining the architecture and design of an information system?
- Which competencies does an information system contain? Give examples of these competencies.