COSC 3380
Design of Database Systems

Database System
Concepts and Architecture

February 7, 2024

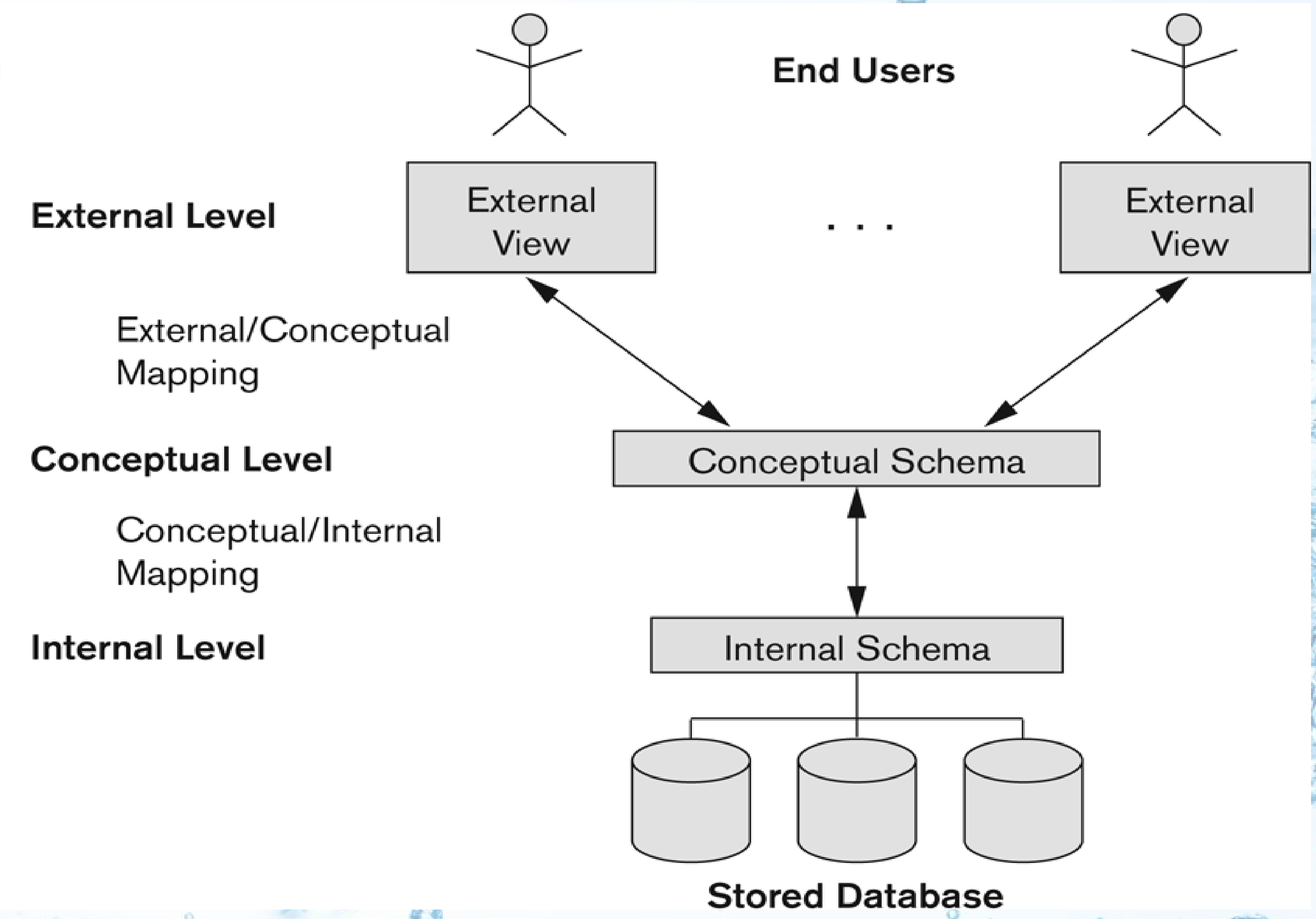
Database Schema vs. Database State

- Distinction
  - The ***database schema*** changes very infrequently
  - The ***database state*** changes every time the data in the database are updated

Three-Schema Architecture

- To support DBMS characteristics of
  - **Program-data independence**
  - Support of **multiple views** of the data
- Defines DBMS schemas at ***three*** levels:
  - **Internal schema** - to describe physical storage structures and access paths (eg. indexes).
    - Typically uses a **physical** data model.
  - **Conceptual schema** - to describe the structure and constraints for the whole database for a community of users.
    - Uses a **conceptual** or an **implementation** data model.
  - **External schema** - to describe the various user views.
    - Usually uses the same data model as the conceptual schema.

Three-Schema Architecture



Three-Schema Architecture

- Mappings among schema levels are needed to transform requests and data.
  - Programs refer to an external schema, and are mapped by the DBMS to the internal schema for execution
  - Data extracted from the internal DBMS level is reformatted to match the user’s external view
  - Example: formatting the results of an SQL query for display in a Web page

Data Independence

- **Logical Data Independence**:
  - The capacity to change the conceptual schema without having to change the external schemas and their associated application programs
  - Example: Adding data items, change constraints
- **Physical Data Independence**:
  - The capacity to change the internal schema without having to change the conceptual schema
  - Example: Reorganizing file structures, creating new indexes to improve database performance
- **Which one is easier to achieve?**

Data Independence

- When a schema at a lower level is changed, only the **mappings** between this schema and higher-level schemas need to be changed in a DBMS that fully supports data independence
- The higher-level schemas themselves are **unchanged**.
  - The application programs need not be changed since they refer to the external schemas.

DBMS Languages

- **Data Definition Language (DDL):**
  - Used by the DBA and database designers to specify the conceptual schema of a database
  - In many DBMSs, the DDL is also used to define internal and external schemas (views)
  - In some DBMSs, separate **storage definition language (SDL)** and **view definition language (VDL)** are used to define internal and external schemas
    - SDL is typically realized via DBMS commands provided to the DBA and database designers

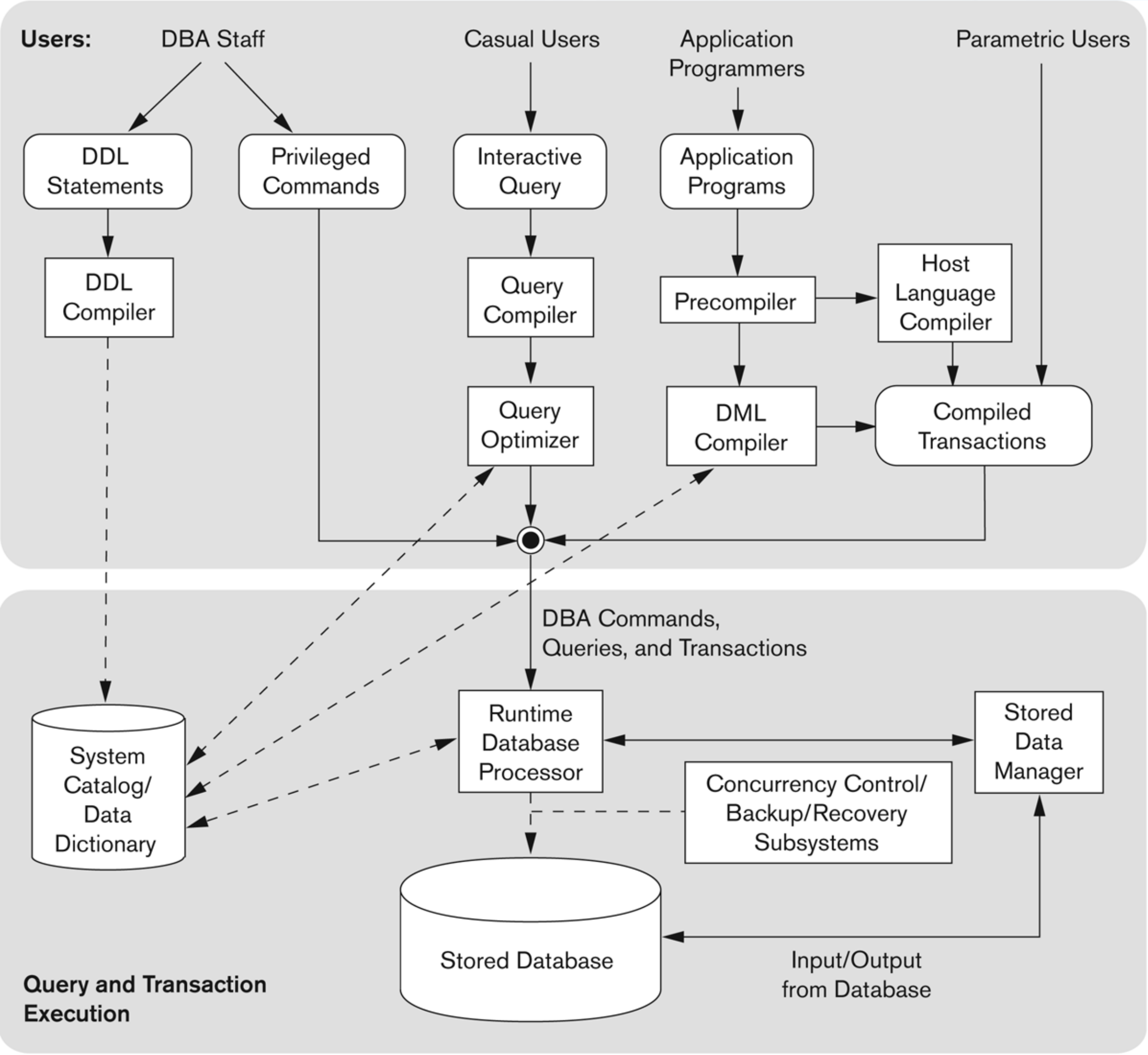
DBMS Languages

- **Data Manipulation Language (DML)**
  - Used to specify database retrievals and updates
  - DML commands (data sublanguage) can be *embedded* in a general-purpose programming language (host language), such as C, C++, C#, or Java
    - A library of functions can also be provided to access the DBMS from a programming language
  - Alternatively, stand-alone DML commands can be applied directly (*query language*)

Types of DML

- **High Level or Non-procedural Language**
  - For example, the SQL relational language
  - Are “set”-oriented and specify what data to retrieve rather than how to retrieve it
  - A **declarative** language
- **Low Level or Procedural Language**
  - Retrieve data one record-at-a-time
  - Constructs such as looping are needed to retrieve multiple records, along with positioning pointers

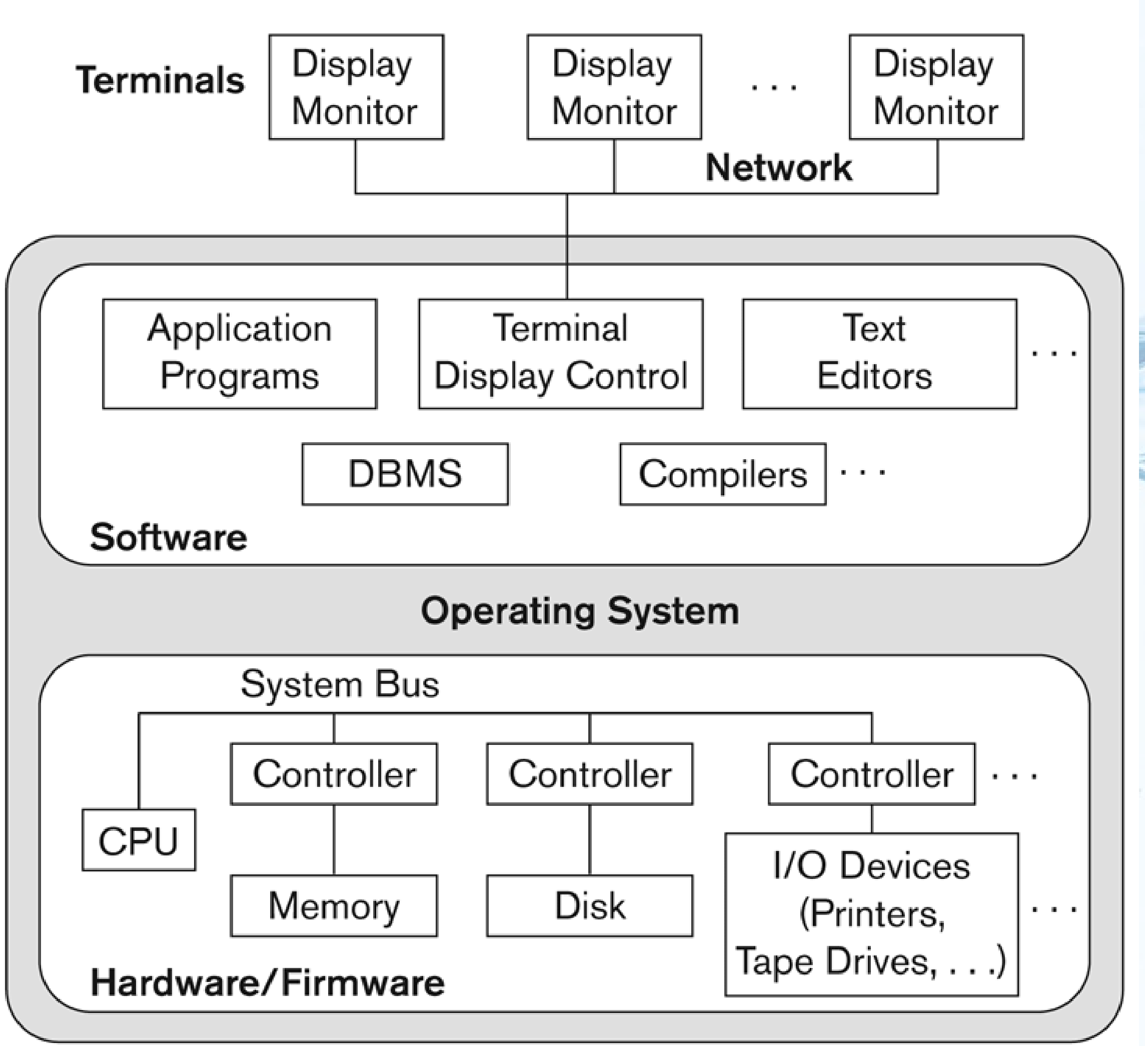
Typical DBMS Modules



DBMS Architectures

- Centralized DBMS
  - Combines everything into single system including
    - DBMS software
    - Hardware,
    - Application programs
    - User interface processing software
  - User can still connect through a remote terminal
    - All processing is done at central system

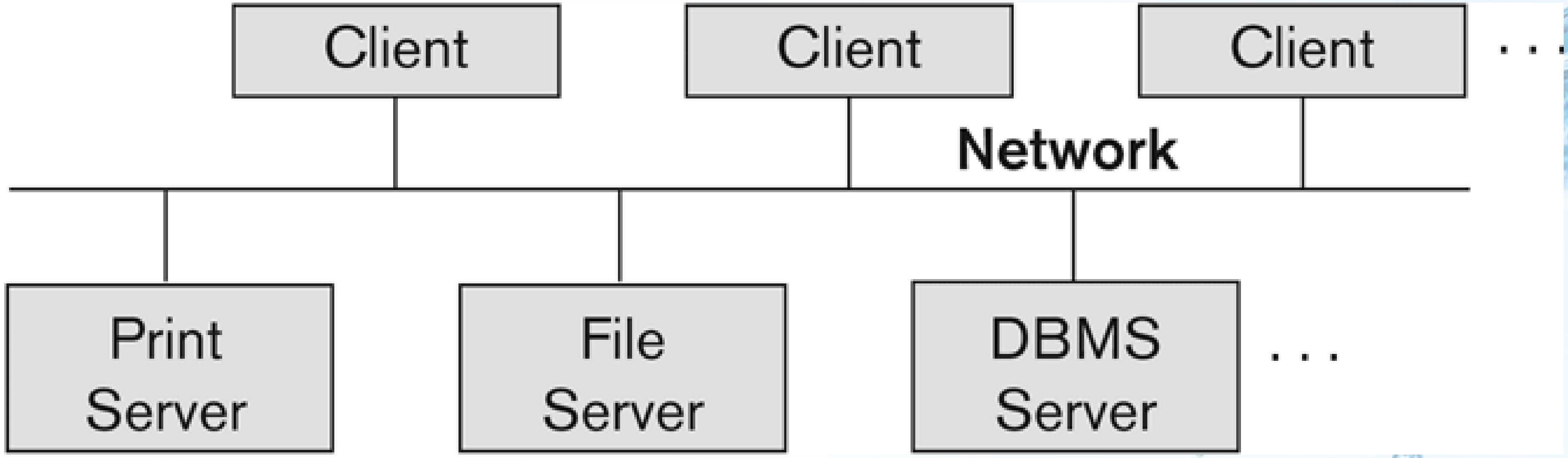
Centralized Architecture



Basic 2-tier Client-Server Architectures

- Specialized Servers with specialized functions
  - Print server
  - File server
  - DBMS server
  - Web server
  - Email server
- Clients can access the specialized servers as needed

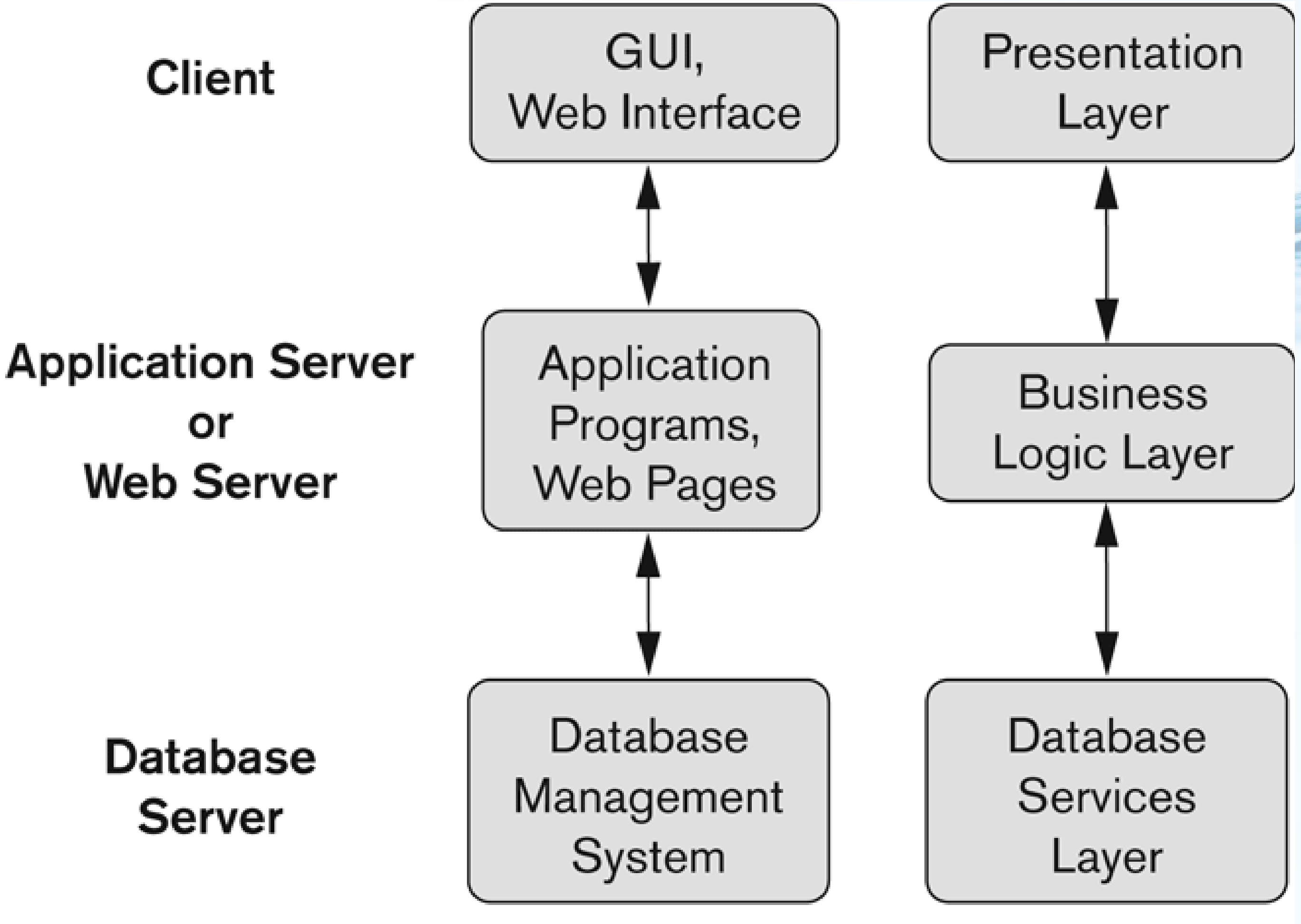
Logical two-tier Client Server Architecture



Three Tier Client-Server Architecture

- Common for Web applications
- Intermediate Layer called Application Server or Web Server
  - Stores the web connectivity software and the business logic part of the application used to access the corresponding data from the database server
  - Acts like a conduit for sending partially processed data between the database server and the client.
- Three-tier Architecture Can Enhance Security
  - Clients cannot directly access database server
  - Database server only accessible via middle tier
  - Client is typically a PC or a mobile device connected to the Web
  - Clients contain user interfaces and Web browsers

Three-tier client-server architecture



Client

GUI, Web Interface

Presentation Layer

Application Server or Web Server

Application Programs, Web Pages

Business Logic Layer

Database Server

Database Management System

Database Services Layer

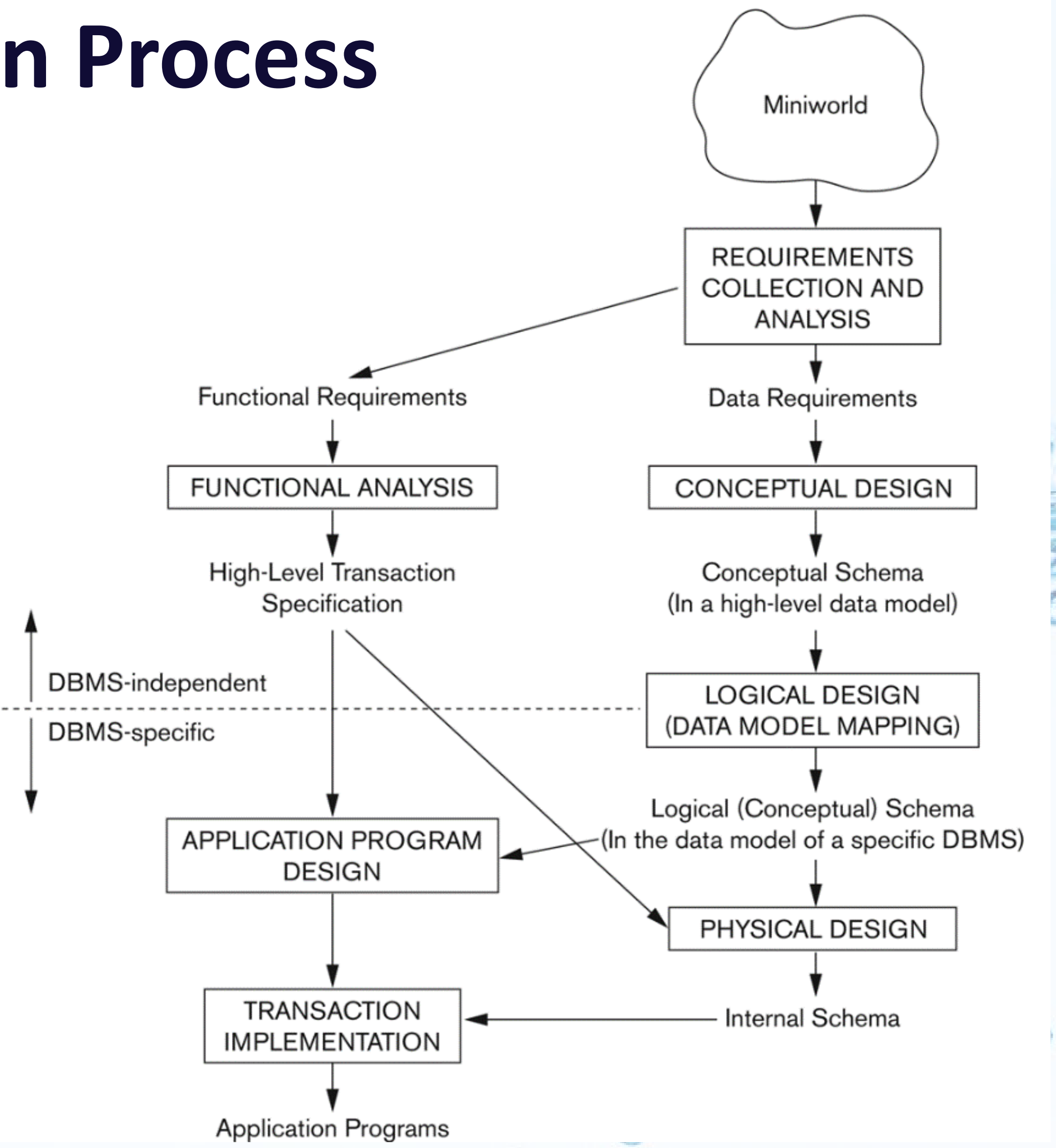
Data Modeling with Entity-Relationship (ER) Model

Name
Employee
ISA
Does
Mechanic
Salesman
Price
RepairJob
Date
Buys
License
Value
Repairs
Car
seller
ufacturer
Model
Year
Name

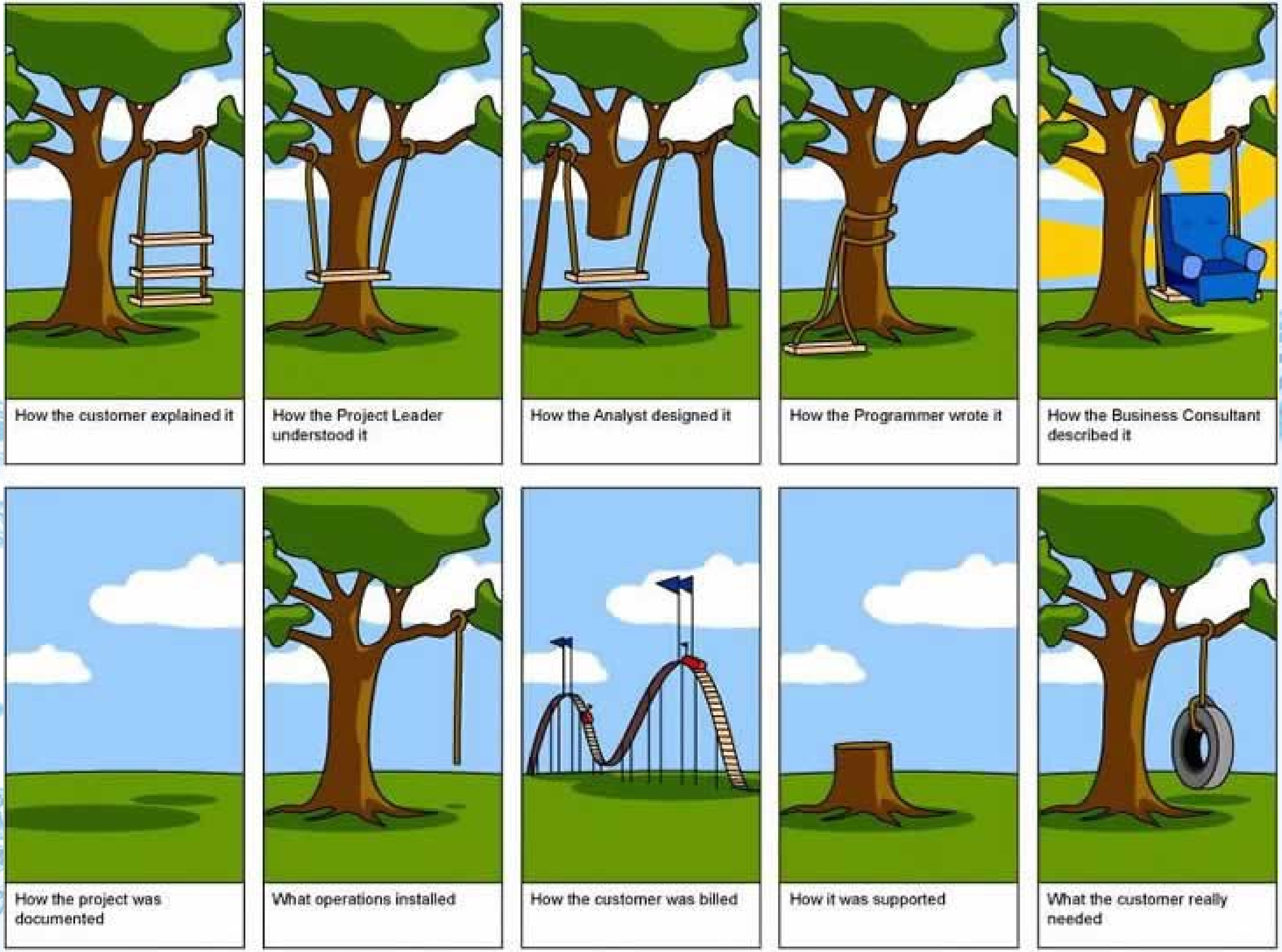
Overview of Database Design Process

- Two main activities
  - Database design
  - Applications design
- Conceptual database design
  - To design the conceptual schema for a database application
- Applications design
  - Focus on the programs and interfaces that access the database
  - Considered as part of software engineering

Overview of Database Design Process



Miniworld
REQUIREMENTS COLLECTION AND ANALYSIS
Functional Requirements
Data Requirements
FUNCTIONAL ANALYSIS
CONCEPTUAL DESIGN
High-Level Transaction Specification
Conceptual Schema (In a high-level data model)
DBMS-independent
DBMS-specific
LOGICAL DESIGN (DATA MODEL MAPPING)
Logical (Conceptual) Schema (In the data model of a specific DBMS)
APPLICATION PROGRAM DESIGN
PHYSICAL DESIGN
TRANSACTION IMPLEMENTATION
Internal Schema
Application Programs


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed