

Chapter 1

Object-Oriented Modeling

Object orientation is a strategy for organizing systems as collections of *interacting objects that combine data and behavior*. It applies to many areas including:

- Software Engineering
- Database Engineering
- Hardware Design

Software and Database Engineering

Software (*Database*) Engineering is a systematic approach to software (*database*) development that emphasizes thorough conceptual understanding prior to design and coding

Database Engineering

- Database applications tend to have a much larger and more complex data structure and less rich behavior compared to programming applications.

Software Development Process

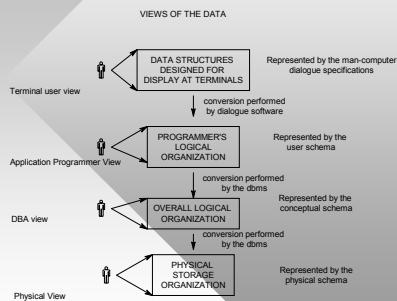
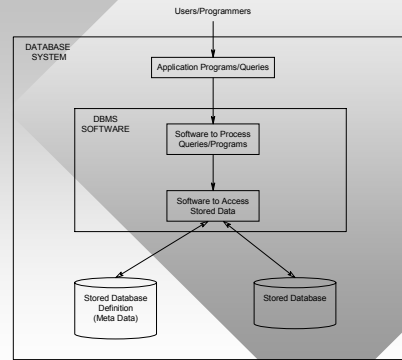
- Conceptualization (conceive an application)
- Analysis (model what not how)
- System Design (design the system architecture)
- Detailed Design (transform model closer to the computer: tables, files, etc. without details of target system)
- Implementation (code)
- Test (does code achieve the requirements?)
- Maintenance (documentation and traceability is important)

Object-Oriented Modeling versus Object-Oriented Programming

- Object-Oriented Programming (OOPS: Java, C++, Smalltalk, CLOS, etc.) is emphasized in the later stages of the software life cycle (implementation)
- Object-Oriented Modeling is important throughout the entire design life cycle from analysis through design, implementation, test, and maintenance

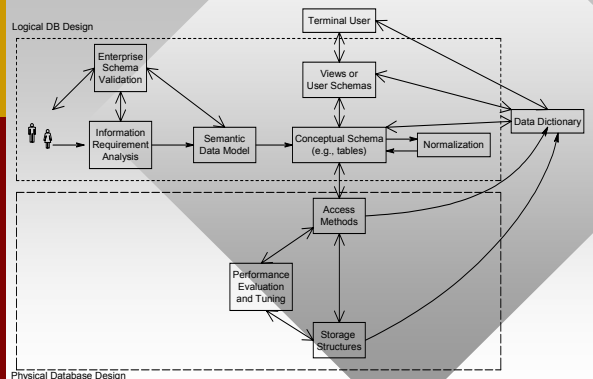
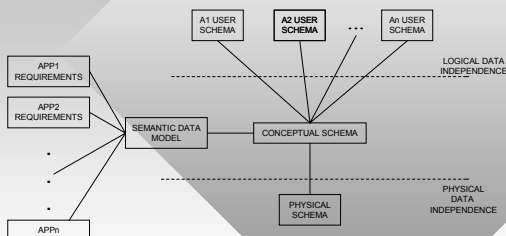
What is a Database?

- A set of data stored on disks or other storage media
- A set of application programs which run against the data with the typical operations such as: retrieve, update, insert, and delete

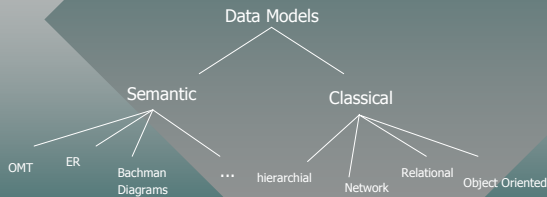


SCHEMAS

- Schema - Map or plan of the structure of a database for a particular view
- Enterprise Schema - pure representation of the real world that is independent of storage and efficiency considerations
- Conceptual Schema - the version of the enterprise schema that can be presented to the database system
- Physical Schema - takes into consideration the distribution of the data, access methods, and indexing techniques
- User Schema - takes into consideration how individual uses view the data



DBMS should support at least one classical data model



Classical - closer to the machine expressed in terms used by the DBMS
Semantic - captures more meaning and better for representing the real world

Abstraction

- Humans use abstraction everyday to focus on the essential aspects of a problem while ignoring details
- Resist the temptation to dive into the implementation details in both software and database design

Develop a good model first

- Requires a thorough understanding of the problem
- Provides a means for deep thinking about designs and a more precise expression of ideas
- Can improve the odds of a successful large-scale project by reducing the overall development time and increasing the flexibility of the design

Semantic-Data Model

- An abstraction of some aspect of a problem expressed with various kinds of diagrams
- Provide the basis for analyzing requirements and guide the design and implementation of a system

Modeling Advantages

- **Reduce life cycle cost** (clear documentation helps maintenance and allows more code reuse)
- **Better quality** (provides a formal means for sharing designs with others for peer and client review)
- **Faster time to market** (facilitates the design of working units that can be delegated)

Modeling Advantages

- **Communication** (models promote communication among all parties by separating deep conceptual issues from implementation details)
- **Extensibility** (Imposes structure on the implementation code which makes changes easier to make)

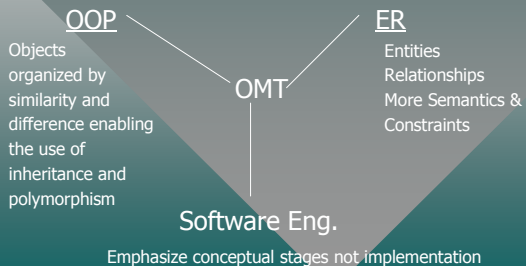
Modeling Advantages

- **Traceability** (tools facilitate the impact of changes in the requirements to the implementation code)
- **Better selection** (models help those who purchase a custom system from a vendor)

Next Generation OMT (Union of UML and OMT)

- OMT is essentially an extended Entity-Relationship (ER) model
- Booch, Rumbaugh, and Jacobson have developed a new notation that unifies the Booch, OMT, and Objectory notations (UML)
- UML emphasizes programming apps while our text emphasizes next generation OMT for DB apps

OMT Origins



OMT Models

