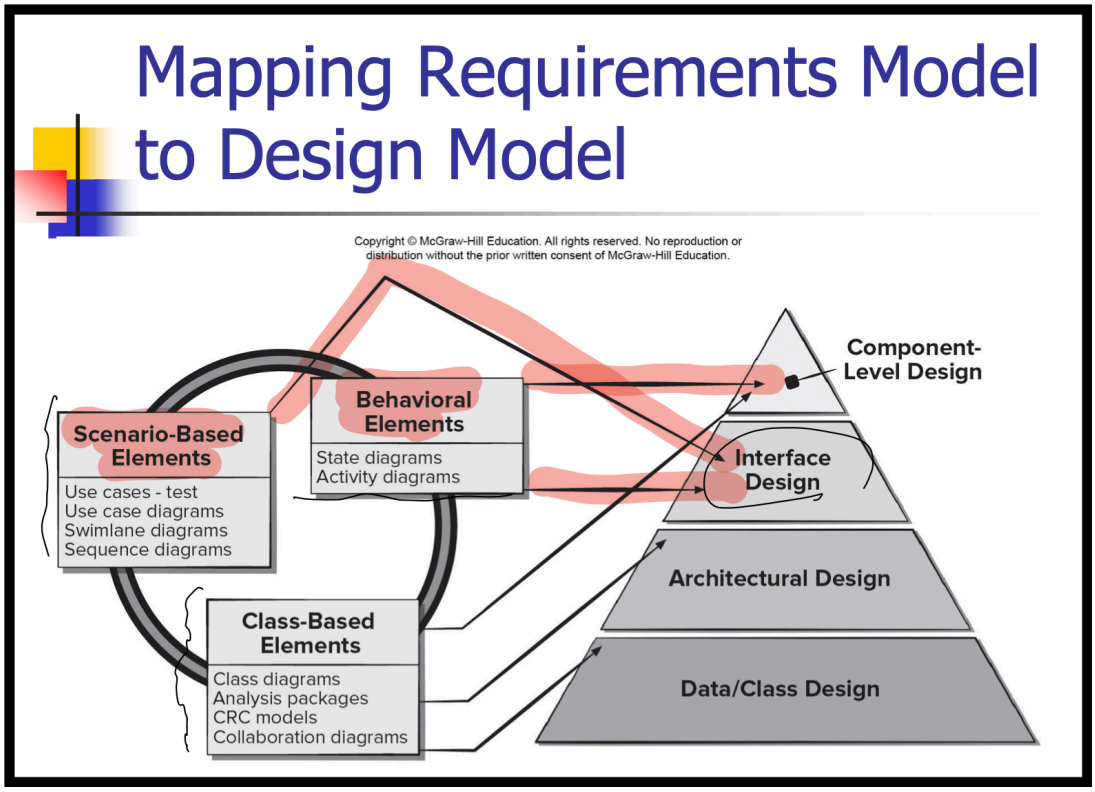
6 Design Concepts: Chapter 9 of the Textbook.

Software Design:

- Set of principles, concepts, and practices that lead to the development of a high quality system or product.
- Design concepts must be understood before the machines applied.

Software Engineering Design:

1. Data/class design - analysis classes into implementation classes and data structures.
2. Architectural design - relationships among the major software structural elements.
3. Interface design - defines software elements, hardware , and end-users communicate.
4. Component level design - structural elements into procedural descriptions of software component.



Design and Quality

- Design must implement all of the explicit requirements contained in the analysis model and it must accommodate all of the implicit requirements desired by the customer.
- Design should be readable and easy to understand to guide for people who writes code and test.

- design should provide a complete picture of the software, addressing the data, functional, and behavioural domains.

Common design characteristics - each design introduces unique heuristics and notions

- mechanism for the translating the requirements model into a design representation
- Notation for representing functional components and their interfaces
- Heuristics for refinement and partitioning
- Guidelines for quality assessment

Design concepts
Abstraction - dat, procedural
Architecture - overall structure of organization of software components ways components interact and structure of data used by components
Design patterns - describe a design structure that solves a well defined design problem within a specific context
Separation of concerns - any complex problem can be easily handled if it subdivided into pieces
Modularity - compartmentalization of data and function

Design Concepts

- Information Hiding - controlled interfaces which define and enforces access to component procedural detail and any local data structure used by the component.
- Functional independence - single-minded (high cohesion) components with aversion to excessive interaction with other components (low coupling).
- Stepwise Refinement – incremental elaboration of detail for all abstractions.
- Refactoring—a reorganization technique that simplifies the design without changing functionality.
- Design Classes—provide design detail that will enable analysis classes to be implemented.

Design Class Characteristics

- Complete - includes all necessary attributes and methods) and sufficient (contains only those methods needed to achieve class intent).
- Primitiveness – each class method focuses on providing one service.
- High cohesion – small, focused, single- minded classes.
- Low coupling – class collaboration kept to minimum.

Information Hiding - reduce side effects, limits the global impact of local design decisions, emphasizes communication through controlled interfaces.

Architecture properties:

- Structural properties. This aspect of the architectural design representation defines the components of a system (for example, modules, objects, filters) and the manner components are packaged and interact with one another.
- Extra-functional properties. The architectural design description should address how the design architecture achieves requirements for performance, capacity, reliability, security, adaptability, and other characteristics.
- Families of related systems. The architectural design
should draw upon repeatable patterns (building blocks)
often encountered in the design of similar systems.

Data Design Elements

- Data model – data objects and database architectures.
  - Examines data objects independently of processing.
  - Focuses attention on the data domain.
  - Creates a model at the customer's level of abstraction.
  - Indicates how data objects relate to one another.

- Data object can be an external entity, a thing, an event, a place, a role, an organizational unit, or a structure.
- Data objects contain a set of attributes that act as a quality, characteristic, or descriptor of the object.

- Data objects may be connected to one another in many different ways.

Architectural Design Elements

The architectural model is derived from three sources:
- Information about the application domain for the software to be built.
- Specific requirements model elements such as data flow analysis classes and their relationships (collaborations) for the problem at hand, and
- Availability of architectural patterns and styles

Interface Design Elements

- Interface is a set of operations that describes the externally observable

behaviour of a class and provides access to its public operations. - Tablet and smartphone
- Important elements:
- User interface (UI).
- External interfaces to other systems.
- Internal interfaces between various design components.
- UI or User Experience (UX) is a major engineering action to ensure the creation on usable software products.
- Internal and external interfaces should incorporate - both error checking and appropriate security features.

Component-Level Design Elements

- Describes the internal detail of each software component.
- Defines:
- Data structures for all local data objects.
- Algorithmic detail for all component processing functions. - Interface that allows access to all component operations.
- Modeled using U M L component diagrams.

Deployment Design Elements

- Indicates how software functionality and subsystems will be allocated within the physical computing environment.
- Modeled using UML deployment diagrams.
    - Descriptor form deployment diagrams - show the computing environment but does not indicate configuration details.
    - Instance form deployment diagrams - identify specific hardware configurations and are developed in the latter stages of design.