



Chapter 3: DESIGN STRATEGIES

By:
Ts. Azaliza binti Zainal



ROLES OF SOFTWARE DESIGNERS




■ Systems Engineer.

- Designs systems using a holistic approach, which includes designing how software, hardware, people, etc. collaborate to achieve the system's goal.


■ Software Architect.

- Design software systems using (for the most part) a black-box modeling approach; concern is placed on the external properties of software components that determine the system's quality and support the further design of functional requirements.


ROLES OF SOFTWARE DESIGNERS

- 
- **Component Designer.**
 - Focuses on designing the internal structure and behavior of software components identified during the architecture phase; typically, these designers have strong programming skills, since they implement their designs in code.
 - **User Interface Designer.**
 - Design the software's user interface; skilled in determining ways that increase the usability of the system.


DESIGN STRATEGIES


- 
- A vertical decorative strip on the left side of the slide contains several images: a person at a computer, a globe with binary code, a close-up of a computer keyboard, and a series of four colored dots (blue, green, yellow, red) at the bottom.
- Software design is a process to conceptualize the software requirements into software implementation.
 - Software design takes the user requirements as challenges and tries to find optimum solution.
 - While the software is being conceptualized, a plan is chalked out to find the best possible design for implementing the intended solution.

STRUCTURED DESIGN STRATEGY

- 
- Structured design is a conceptualization of problem into several well-organized elements of solution.
 - It is basically concerned with the solution design.
 - Benefit of structured design is, it gives better understanding of how the problem is being solved.
 - Structured design also makes it simpler for designer to concentrate on the problem more accurately.


STRUCTURED DESIGN STRATEGY

- 
- Structured design is mostly based on 'divide and conquer' strategy where a problem is broken into several small problems and each small problem is individually solved until the whole problem is solved.
 - A good structured design always follows some rules for communication among multiple modules, namely -
 - ❖ **Cohesion** - grouping of all functionally related elements.
 - ❖ **Coupling** - communication between different modules.
 - A good structured design has **high cohesion and low coupling** arrangements.

- 
- In function-oriented design, the system is comprised of many smaller sub-systems known as functions. These functions are capable of performing significant task in the system. The system is considered as top view of all functions.
 - Function oriented design inherits some properties of structured design where divide and conquer methodology is used.

Design Process:

- The whole system is seen as how data flows in the system by means of data flow diagram.
- DFD depicts how functions changes data and state of entire system.
- The entire system is logically broken down into smaller units known as functions on the basis of their operation in the system.
- Each function is then described at large.

- 
- A vertical decorative strip on the left side of the slide. It contains a collage of images: a person at a computer, a globe with a chain, binary code (0s and 1s), a close-up of a computer keyboard, and a row of five blue dots.
- Object oriented design works around the entities and their characteristics instead of functions involved in the software system.
 - This design strategies focuses on entities and its characteristics.
 - The whole concept of software solution revolves around the engaged entities.

Important concepts of Object Oriented Design:

- **Objects** - All entities involved in the solution design are known as objects.
- **Classes** - A class is a generalized description of an object.
- **Encapsulation** - In OOD, the attributes data variables and methods operation on the data are bundled together is called encapsulation.
- **Inheritance** - OOD allows similar classes to stack up in hierarchical manner where the lower or sub-classes can import, implement and re-use allowed variables and methods from their immediate super classes.
- **Polymorphism** - allows a single interface performing tasks for different types.

Software design process can be perceived as series of well-defined steps. Though it varies according to design approach (function oriented or object oriented, yet It may have the following steps involved:


- A solution design is created from requirement or previous used system and/or system sequence diagram.
- Objects are identified and grouped into classes on behalf of similarity in attribute characteristics.
- Class hierarchy and relation among them is defined.
- Application framework is defined.

Top Down Design

- A system is composed of more than one sub-systems and it contains a number of components. Further, these sub-systems and components may have their own set of sub-system and components and creates hierarchical structure in the system.
- Top-down design starts with a generalized model of system and keeps on defining the more specific part of it. When all components are composed the whole system comes into existence.
- Top-down design is more suitable when the software solution needs to be **designed from scratch** and specific details are unknown.

Bottom-up Design

- The bottom up design model starts with most specific and basic components. It proceeds with composing higher level of components by using basic or lower level components. It keeps creating higher level components until the desired system is not evolved as one single component. With each higher level, the amount of abstraction is increased.
- Bottom-up strategy is more suitable when a system needs to be **created from some existing system**, where the basic primitives can be used in the newer system.
- Both, top-down and bottom-up approaches are not practical individually. Instead, a good combination of both is used.

- 
- A vertical decorative strip on the left side of the slide. It contains a collage of images: a person at a computer, a globe with binary code, a close-up of a computer keyboard, and a series of four blue dots at the bottom.
- Design for minimizing complexity.
 - Design is about minimizing complexity.
 - Every decision that is made during design must take into account reducing complexity.
 - When faced with competing design option, always choose the one that minimizes complexity.
 - Design for change.
 - Software will change, design with extension in mind.
 - A variety of techniques can be employed through the design phase to achieve this.

THANK YOU FOR YOUR ATTENTION

By:

Ts. Azaliza Zainal

Software Engineering Cluster, Computing Department, FCVAC