

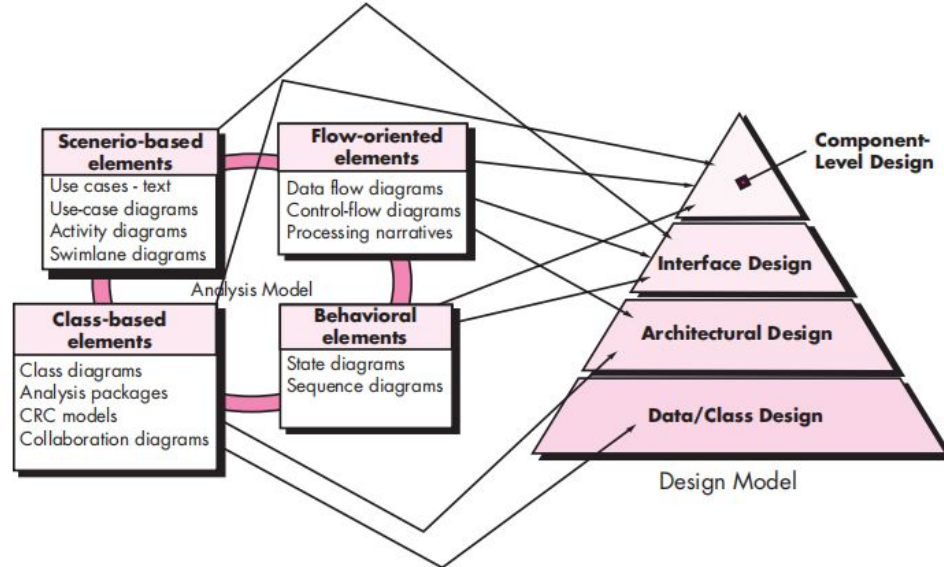
Design Engineering

Module 4

Design Engineering

Design Model

Software design sits at the technical kernel of software engineering and is applied regardless of the software process model that is used. Beginning once software requirements have been analyzed and modeled, software design is the last software engineering action within the modeling activity and sets the stage for construction (code generation and testing)



Design Engineering

Each of the elements of the requirements model provides information that is necessary to create the four design models required for a complete specification of design

The data/class design transforms class models into design class realizations and the requisite data structures required to implement the software. The objects and relationships defined in the CRC diagram and the detailed data content depicted by class attributes and other notation provide the basis for the data design action. Part of class design may occur in conjunction with the design of software architecture. More detailed class design occurs as each software component is designed.

Design Engineering

Each of the elements of the requirements model provides information that is necessary to create the four design models required for a complete specification of design

The data/class design transforms class models into design class realizations and the requisite data structures required to implement the software. The objects and relationships defined in the CRC diagram and the detailed data content depicted by class attributes and other notation provide the basis for the data design action. Part of class design may occur in conjunction with the design of software architecture. More detailed class design occurs as each software component is designed.

Design Engineering

The architectural design defines the relationship between major structural elements of the software, the architectural styles and design patterns that can be used to achieve the requirements defined for the system, and the constraints that affect the way in which architecture can be implemented. The architectural design representation—the framework of a computer-based system—is derived from the requirements model

The interface design describes how the software communicates with systems that interoperate with it, and with humans who use it. An interface implies a flow of information (e.g., data and/or control) and a specific type of behavior. Therefore, usage scenarios and behavioral models provide much of the information required for interface design

Design Engineering

THE DESIGN PROCESS

Software design is an iterative process through which requirements are translated into a “blueprint” for constructing the software. Initially, the blueprint depicts a holistic view of software. That is, the design is represented at a high level of abstraction—a level that can be directly traced to the specific system objective and more detailed data, functional, and behavioral requirements. As design iterations occur, subsequent refinement leads to design representations at much lower levels of abstraction. These can still be traced to requirements, but the connection is more subtle.

Design Engineering

Throughout the design process, the quality of the evolving design is assessed with a series of technical reviews. McGlaughli suggests three characteristics that serve as a guide for the evaluation of a good design:

- The design must implement all of the explicit requirements contained in the requirements model, and it must accommodate all of the implicit requirements desired by stakeholders.
- The design must be a readable, understandable guide for those who generate code and for those who test and subsequently support the software.
- The design should provide a complete picture of the software, addressing the data, functional, and behavioral domains from an implementation perspective.

Design Engineering

Quality attributes

The attributes of design name as 'FURPS' are as follows:

Functionality:

It evaluates the feature set and capabilities of the program.

Usability:

It is accessed by considering the factors such as human factor, overall aesthetics, consistency and documentation.

Reliability:

It is evaluated by measuring parameters like frequency and security of failure, output result accuracy, the mean-time-to-failure(MTTF), recovery from failure and the the program predictability.

Design Engineering

Performance:

It is measured by considering processing speed, response time, resource consumption, throughput and efficiency.

Supportability:

It combines the ability to extend the program, adaptability, serviceability. These three terms define the maintainability.

Testability, compatibility and configurability are the terms using which a system can be easily installed and found the problem easily.

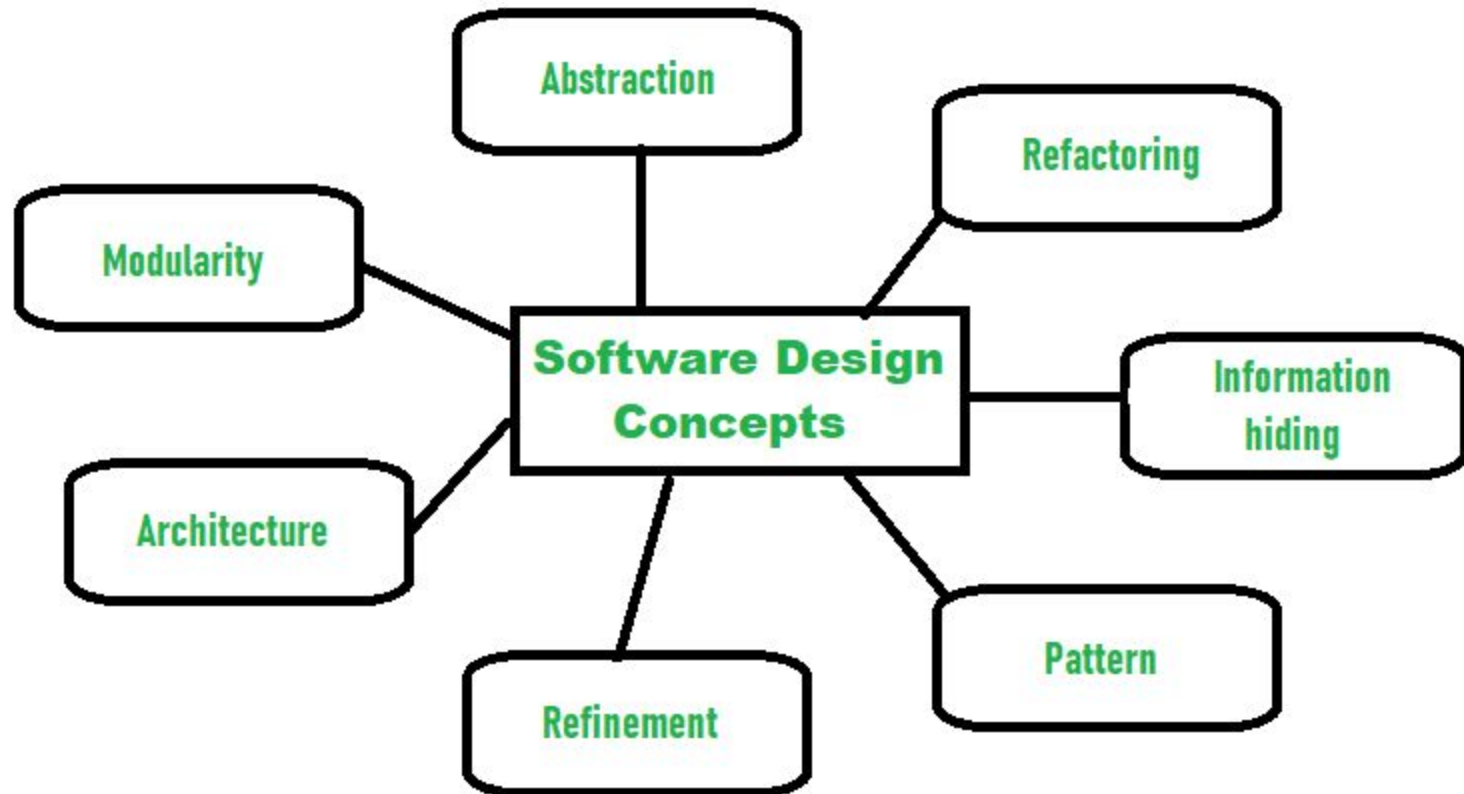
Supportability also consists of more attributes such as compatibility, extensibility, fault tolerance, modularity, reusability, robustness, security, portability, scalability.

Design Engineering

Software Design Concepts:

Concepts are defined as a principal idea or invention that comes into our mind or in thought to understand something. The software design concept simply means the idea or principle behind the design. It describes how you plan to solve the problem of designing software, the logic, or thinking behind how you will design software. It allows the software engineer to create the model of the system or software or product that is to be developed or built. The software design concept provides a supporting and essential structure or model for developing the right software. There are many concepts of software design and some of them are given below:

Design Engineering



Design Engineering

Abstraction- hide Irrelevant data

Abstraction simply means to hide the details to reduce complexity and increases efficiency or quality. Different levels of Abstraction are necessary and must be applied at each stage of the design process so that any error that is present can be removed to increase the efficiency of the software solution and to refine the software solution. The solution should be described in broad ways that cover a wide range of different things at a higher level of abstraction and a more detailed description of a solution of software should be given at the lower level of abstraction.

Design Engineering

Modularity- subdivide the system

Modularity simply means dividing the system or project into smaller parts to reduce the complexity of the system or project. In the same way, modularity in design means subdividing a system into smaller parts so that these parts can be created independently and then use these parts in different systems to perform different functions. It is necessary to divide the software into components known as modules because nowadays there are different software available like Monolithic software that is hard to grasp for software engineers. So, modularity in design has now become a trend and is also important. If the system contains fewer components then it would mean the system is complex which requires a lot of effort (cost) but if we are able to divide the system into components then the cost would be small.

Design Engineering

Architecture- design a structure of something

Architecture simply means a technique to design a structure of something. Architecture in designing software is a concept that focuses on various elements and the data of the structure. These components interact with each other and use the data of the structure in architecture.

Refinement- removes impurities

Refinement simply means to refine something to remove any impurities if present and increase the quality. The refinement concept of software design is actually a process of developing or presenting the software or system in a detailed manner that means to elaborate a system or software. Refinement is very necessary to find out any error if present and then to reduce it.

Design Engineering

Pattern- a repeated form

The pattern simply means a repeated form or design in which the same shape is repeated several times to form a pattern. The pattern in the design process means the repetition of a solution to a common recurring problem within a certain context.

Information Hiding- hide the information

Information hiding simply means to hide the information so that it cannot be accessed by an unwanted party. In software design, information hiding is achieved by designing the modules in a manner that the information gathered or contained in one module is hidden and can't be accessed by any other modules.

Design Engineering

Refactoring- reconstruct something

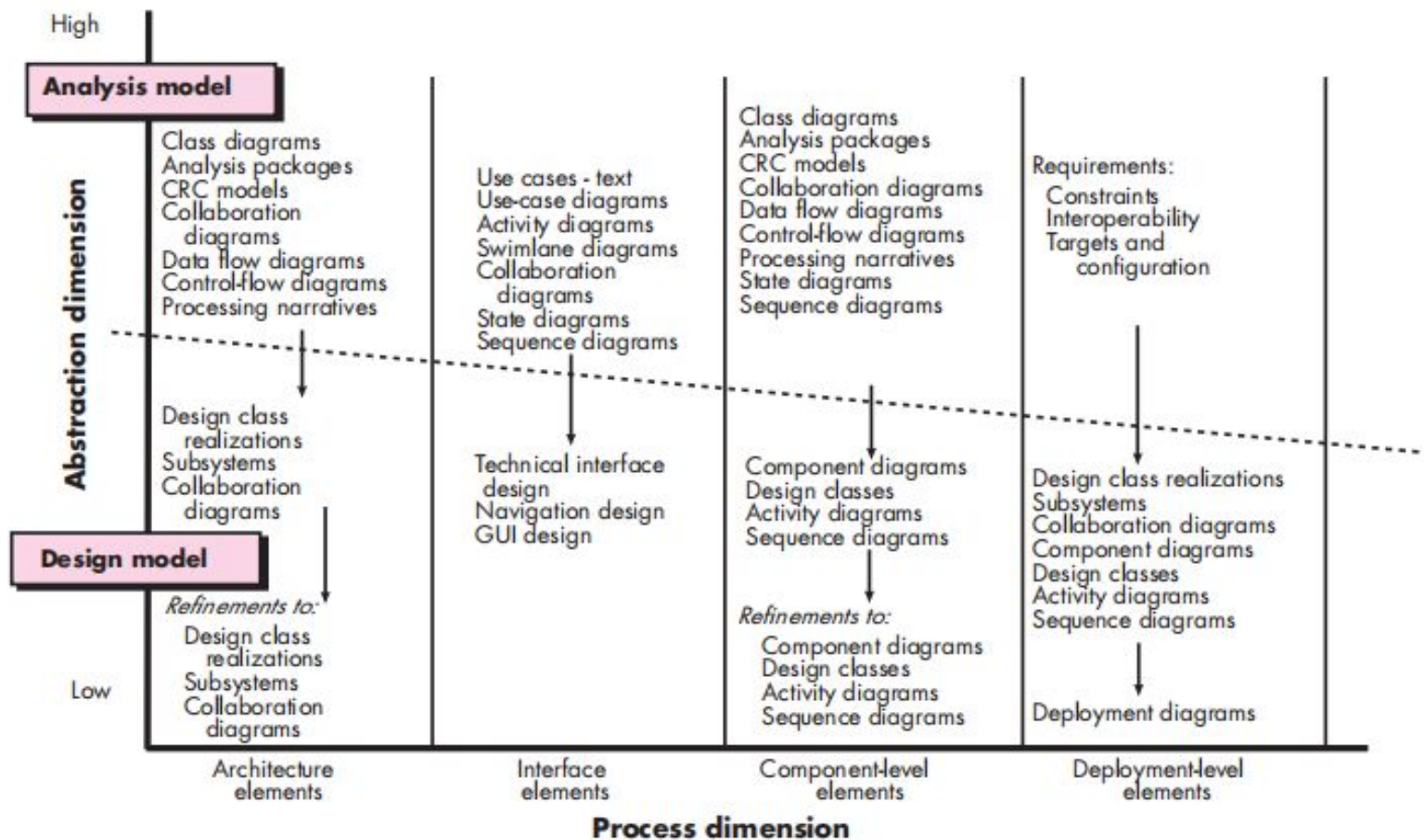
Refactoring simply means reconstructing something in such a way that it does not affect the behavior of any other features. Refactoring in software design means reconstructing the design to reduce complexity and simplify it without affecting the behavior or its functions. Fowler has defined refactoring as “the process of changing a software system in a way that it won’t affect the behavior of the design and improves the internal structure”.

Design Engineering

THE DESIGN MODEL in elaboration:

The design model can be viewed in two different dimensions as illustrated in Figure. The process dimension indicates the evolution of the design model as design tasks are executed as part of the software process. The abstraction dimension represents the level of detail as each element of the analysis model is transformed into a design equivalent and then refined iteratively.

Design Engineering



Design Engineering

Data Design Elements

Like other software engineering activities, data design (sometimes referred to as data architecting) creates a model of data and/or information that is represented at a high level of abstraction (the customer/user's view of data). This data model is then refined into progressively more implementation-specific representations that can be processed by the computer-based system. In many software applications, the architecture of the data will have a profound influence on the architecture of the software that must process it.

Design Engineering

Architectural Design Elements

The architectural design for software is the equivalent to the floor plan of a house. The floor plan depicts the overall layout of the rooms; their size, shape, and relationship to one another; and the doors and windows that allow movement into and out of the rooms. The floor plan gives us an overall view of the house. Architectural design elements give us an overall view of the software.

Design Engineering

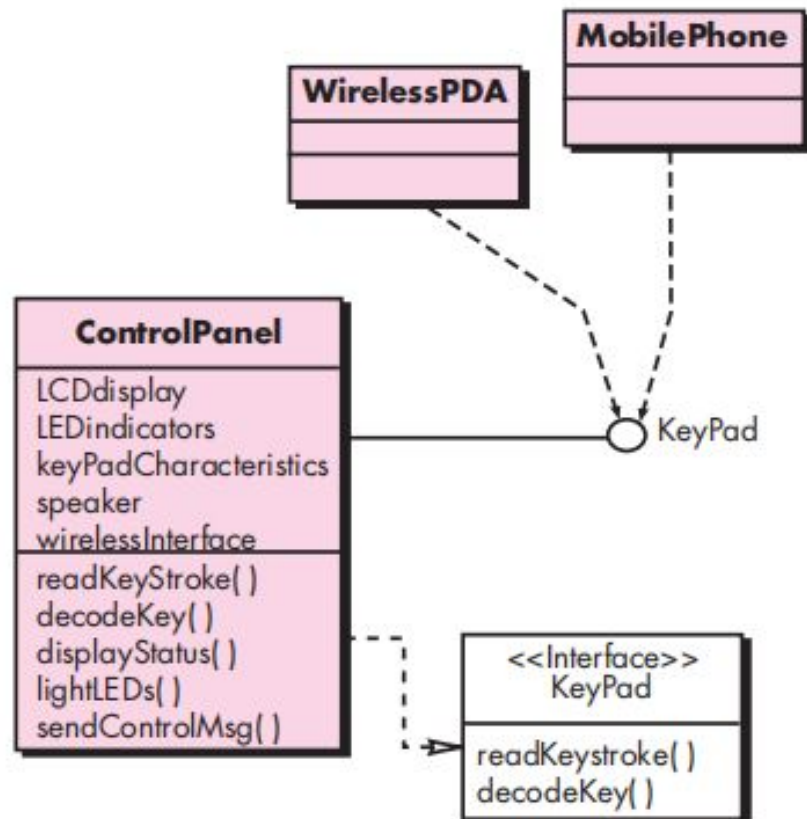
Interface Design Elements:

The interface design elements for software depict information flows into and out of the system and how it is communicated among the components defined as part of the architecture.

There are three important elements of interface design:

- (1) the user interface (UI);
- (2) external interfaces to other systems, devices, networks, or other producers or consumers of information; and
- (3) internal interfaces between various design components. These interface design elements allow the software to communicate externally and enable internal communication and collaboration among the components that populate the software architecture

Design Engineering



Design Engineering

Component-Level Design Elements:

The component-level design defines data structures for all local data objects and algorithmic detail for all processing that occurs within a component and an interface that allows access to all component operations (behaviors). Within the context of object-oriented software engineering, a component is represented in UML diagrammatic form as shown in Figure



Design Engineering

Deployment-Level Design Elements

Deployment-level design elements indicate how software functionality and subsystems will be allocated within the physical computing environment that will support the software. During design, a UML deployment diagram is developed and then refined.

