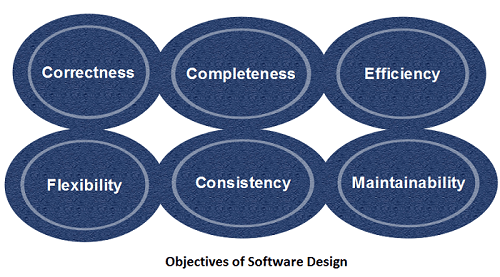
Software Design

* Software design is a mechanism to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.
* It deals with representing the client's requirement, as described in SRS (Software Requirement Specification) document, into a form, i.e., easily implementable using programming language.
* The main aim of design engineering is to generate a model which shows firmness, delight and commodity.
* Software design is an iterative process through which requirements are translated into the blueprint for building the software.

Objectives of Software Design

Following are the purposes of Software design:



1. **Correctness:** Software design should be correct as per requirement.
2. **Completeness:** The design should have all components like data structures, modules, and external interfaces, etc.
3. **Efficiency:** Resources should be used efficiently by the program.
4. **Flexibility:** Able to modify on changing needs.
5. **Consistency:** There should not be any inconsistency in the design.
6. **Maintainability:** The design should be so simple so that it can be easily maintainable by other designers.

Software quality guidelines

* A design is generated using the recognizable architectural styles and compose a good design characteristic of components and it is implemented in evolutionary manner for testing.
* A design of the software must be modular i.e the software must be logically partitioned into elements.
* In design, the representation of data , architecture, interface and components should be distinct.
* A design must carry appropriate data structure and recognizable data patterns.
* Design components must show the independent functional characteristic.
* A design creates an interface that reduce the complexity of connections between the components.
* A design must be derived using the repeatable method.
* The notations should be use in design which can effectively communicates its meaning.

## Design concepts

**The set of fundamental software design concepts are as follows:**

1. **Abstraction**

An abstraction is a tool that enables a designer to consider a component at an abstract level without bothering about the internal details of the implementation. Abstraction can be used for existing element as well as the component being designed.

Here, there are two common abstraction mechanisms

1. Functional Abstraction
2. Data Abstraction

### **Functional Abstraction**

1. A module is specified by the method it performs.
2. The details of the algorithm to accomplish the functions are not visible to the user of the function.

Functional abstraction forms the basis for **Function oriented design approaches**.

### **Data Abstraction**

Details of the data elements are not visible to the users of data. Data Abstraction forms the basis for **Object Oriented design approaches**.

**2. Architecture**

* The complete structure of the software is known as software architecture.
* Structure provides conceptual integrity for a system in a number of ways.
* The architecture is the structure of program modules where they interact with each other in a specialized way.
* The components use the structure of data.
* The aim of the software design is to obtain an architectural framework of a system.
* The more detailed design activities are conducted from the framework.

1. **Patterns**A design pattern describes a design structure and that structure solves a particular design problem in a specified content.
2. **Modularity**

* A software is separately divided into name and addressable components. Sometime they are called as modules which integrate to satisfy the problem requirements.
* Modularity is the single attribute of a software that permits a program to be managed easily.

**5. Information hiding**

Modules must be specified and designed so that the information like algorithm and data presented in a module is not accessible for other modules not requiring that information.

**6. Functional independence**

* The functional independence is the concept of separation and related to the concept of modularity, abstraction and information hiding.
* The functional independence is accessed using two criteria i.e Cohesion and coupling.

**Cohesion**

In computer programming, cohesion defines to the degree to which the elements of a module belong together. **Thus, cohesion measures the strength of relationships between pieces of functionality within a given module. For example, in highly cohesive systems, functionality is strongly related.**

* A cohesive module performs a single task and it requires a small interaction with the other components in other parts of the program.

**Coupling**  
Coupling is an indication of interconnection between modules in a structure of software.

In software engineering, the coupling is the degree of interdependence between software modules. Two modules that are tightly coupled are strongly dependent on each other.

**7.Refactoring**

* It is a reorganization technique which simplifies the design of components without changing its function behaviour.
* Refactoring is the process of changing the software system in a way that it does not change the external behaviour of the code still improves its internal structure.

**8. Design classes**

* The model of software is defined as a set of design classes.
* Every class describes the elements of problem domain and that focus on features of the problem which are user visible.

