Basics - Tutorials point

Tuesday, December 25, 2018 12:57 PM

What is a software design?
Software design is a process to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.

Software design is the first step in Software Design Life Cycle (SDLC), which moves the focus from problem domain to solution domain. It tries to specify how to fulfill the requirements mentioned in SRS (Software Requirement Specification - document to access the user requirements).

What are the different Software Design Levels?
Different software design levels are:
- Architectural Design
- High-level Design
- Detailed Design

Architectural Design:
- Highest abstract version of the system
- Software as a system with many components interacting with each other
- Designers get the idea of proposed solution domain

High-level Design:
- Breaks the single - entity multiple component concept of architectural design into less-abstracted view of sub-systems and modules and depicts their interaction with each other.
- High-level design focuses on how the system along with all of its components can be implemented in the forms of modules.
- Recognizes modular structure of each sub-system and their relation and interaction among each other.

Detailed Design:
- It deals with the implementation part of system and sub-system in the previous two designs
- More detailed towards modules and their implementations.
- Defines logical structure of each module and their interfaces to communicate with other modules.

What is modularization?
- It is a technique to divide a software system into multiple discrete and independent modules, which are expected to carry out task(s) independently.
- They may work as basic constructs for the entire software.
- Designers tend to design modules so that they can be executed and/or compiled separately and independently.
- It unintentionally follows the rules of 'divide and conquer' problem-solving strategy.

What are the advantages of Modularization?
- Smaller components are easier to maintain
- Program can be divided based on functional aspects
- Desired level of abstraction can be brought in the program
- Components with high cohesion can be re-used again
- Concurrent execution can be made possible
- Desired from security aspect

What is concurrency?

Earlier times, software are meant to be executed sequentially. The code instruction will be executed one after another implying only one portion of program being activated at any given time.
Concurrency is implemented by splitting the software into multiple independent units of execution, like modules and executing them in parallel. It gives the capability to execute more than on part of code in parallel to each other.

Example:
The spell check feature in word processor is a software module that can be run alongside the word processor itself.

What is Cohesion?
It is a measure that defines the intra-dependability within elements of a module. The greater the cohesion, the better the program design

What are the different types of cohesion?
Co-incidental Cohesion:
Unplanned and random cohesion, due to the result of breaking the program into smaller modules for the sake of modularization. As it is unplanned it causes confusion.

Logical Cohesion:
Logically categorized elements are put together into a module.

Temporal Cohesion:
When elements of module are organized such that they are processed at a similar point in time.

Procedural Cohesion:
When elements of module are grouped together when they are executed sequentially in order to perform a task.

Communicational Cohesion:
When elements of a module are grouped together, which are executed sequentially and work on same data (information).

Sequential Cohesion:
When elements of module are grouped because the output of one element serves as input to another and so on, it is called sequential cohesion.

Functional Cohesion:
It is the highest degree of cohesion and it is highly expected. Elements of module are grouped because they all contribute to a single well-defined function. It can also be reused.

What is Coupling?
Coupling is a measure that defines the level of inter-dependability among modules of a program.
It indicates at what level the modules interfere and interact with each other.
If the coupling is lower, then the coupling is better.

What are the five levels of coupling?
Content Coupling:
When a module can directly access or modify or refer to the content of another module, it is called content level coupling.

Common Coupling:
When multiple modules have read and write access to some global data, it is called common or global

coupling

Control coupling:
Two modules are called control-coupled if one of them decides the function of the other module or changes its flow of execution.

Stamp coupling:
When multiple modules share common data structure and work on different part of it, it is called stamp coupling.

Data coupling:
Data coupling is when two modules interact with each other by means of passing data (as parameter). If a module passes data structure as parameter, then the receiving module should use all its components.

No coupling is considered to be the best.

What are the outputs of the Software Design Process?

- Design Documentation
- Pseudo Codes
- Detailed logic diagrams
- Process diagrams
- Detailed description of all functional or non-functional requirements

Software implementation phase depends on all of them.

What is a design verification and why it is important?
It is necessary to verify the output before proceeding to the next phase. The mistakes that are detected earlier can be solved easily or it will become undetected until testing of the product. A thorough design review can be used for verification and validation.
Using structured verification approach, reviewers can detect the defects that might be caused.

Reference: https://www.tutorialspoint.com/software_engineering/software_design_basics.htm