Software Engineering

→ Modularisation : It is a technique to divide a software system into multiple discrete & independent modules which are expected to be capable of carrying out the work independently.
Modules can be considered as the basic components of the software & being independent, they can be executed separately. This process follows divide-and-conquer problem solving statergy.

⇒ ADVANTAGES OF MODULARISATION

i) Breaking the entire problem into modules makes the process easy to understand, develop and maintain.

ii) With this process, features like Object oriented concepts, cohesion, coupling & reusability can be implemented easily.

(iii) concurrent development & execution of modules makes the software dev. process easier, stable and faster.

(iv) This process facilitates abstractions & levels for understanding the requirement and working of the software.

→ Concurrency : In SE, concurrency is implemented by splitting the software into multiple independent units of execution similar to the modules that are getting executed parallely. Concurrency provides capability to the software to execute more than one part of code in parallel making the execution, integration and testing process more stable & faster.
For example, spell ~~or~~ checker module runs concurrently with the word processor, facilitating grammer & spelling check during preparation of document.

→ Relationship among modules.

Modules have a characteristic to be independent but being part of a S/w, they must be associated with the other modules with the same S/w.

The characteristics of being independent but still related is important for any software. These characteristics are studied under cohesion & coupling.

→ Cohesion : It is a property that defines & measures degree of intra-dependability within elements of a model. The greater the cohesion, the better is the program design. In other words, this property reflects the independentness of a module. ~~The~~ However total independentness of the module is not allowed since these are the components of the software.

→ 7 types of cohesion:-

i) Co-incidental cohesion — RANDOM
It is unplanned and random cohesion which might be due to breaking of this program into smaller modules for the sake of modularisation. Since it is unplanned, it has confusion and inconsistency that creates problem for the programmers.

ii) Logical cohesion — LOGICAL
If the cohesion is made on the basis of logic of individual module such that the module seems to be logically independent, it is known as logical cohesion.

iii) Temporal cohesion — TIME BASED
When elements of module are organised such that they are processed at a similar point of time, it is called temporal cohesion.

iv) Procedural cohesion — Executed Sequencially
When elements of a module are grouped together which are executed sequencially in order to perform a task, it is called procedural cohesion.

Executed Sequencially & Data
v) Communicational cohesion —
When elements of module are grouped together which are executed sequencially and work on same data / information is called communicational cohesion.

vi) Sequencial cohesion —
When elements of module are grouped because the output of 1 element serve as the input to another & so on, it is called sequencial cohesion.

vii) Functional cohesion — Based on function
This is considered to be a highest degree of cohesion & it is highly expected. Elements of the module

in functional cohesion are grouped as they all contribute to a single well-defined function. This type of cohesions supports reusability.

→ Coupling –
It is a measure that defines the level of inter-dependability among modules of a program. This property explains the levels of module interfaces & how they interact with each other. The lower the coupling, better is the program. In other words, this property makes a bonding b/w the different modules of the program.

→ 5 levels of coupling –

i) Content coupling –
When a module can directly access or modify or refer to the content of another module, it is called content level coupling.

ii) Common coupling –
When multiple modules have read & write access to some global data, it is called common/global coupling.

iii) Controlled coupling –
2 modules are called controlled coupling if one of them decides the func. of the other module or changes its flow of execution.

iv) Stamp coupling –
When multiple modules share common data structure & work on diff. part of it, it is called stamp coupling.

v) Data coupling –
When 2 modules interact with each other by means of passing data as parameters, it is known as data coupling. If a module passes data structure as parameter then the receiving module should use all its

component. Ideally, no coupling is considered to be the best & there must be very less coupling b/w the modules.