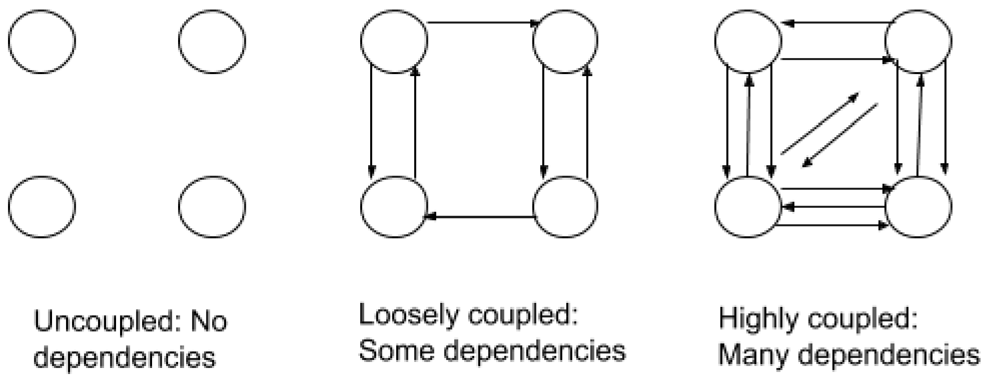
Assignment 7: Cohesion and Coupling

E/15/119
E/15/202
E/15/208

Question 1

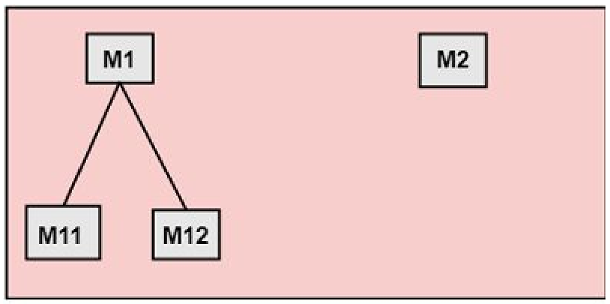
Explain about different types of cohesion and coupling in software engineering

Coupling is the measure of the degree of interdependence between software modules. Two modules that are tightly coupled are strongly dependent on each other. In contrast, two modules that are loosely coupled are not dependent on each other. Uncoupled modules have no interdependence at all within them. However, a good software will have low coupling. That means low coupling is often a sign of a well- structured computer system and a good design. Coupling is usually contrasted with cohesion. Low coupling often correlates with high cohesion, and high coupling often correlates with low cohesion.


Uncoupled: No dependencies
Loosely coupled: Some dependencies
Highly coupled: Many dependencies

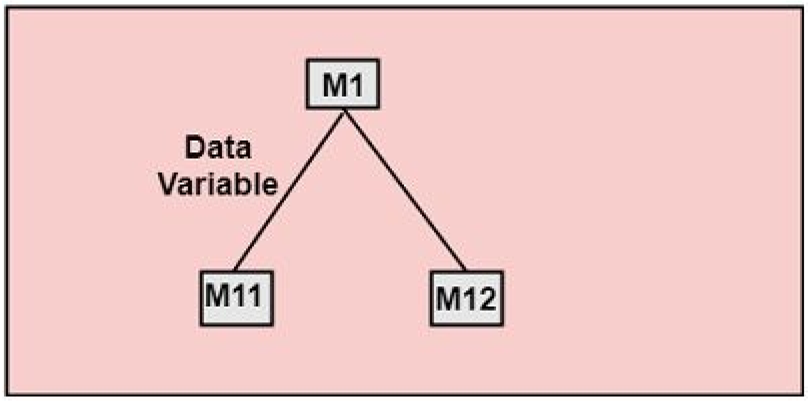
There are number of coupling types:

- No Direct Coupling
    - ❑ There is no direct coupling between modules.
      For example consider the below figure



There is no direct coupling between M1 and M2
In this case modules are subordinates to different modules.

- Data Coupling
    - ❑ Modules are said to be data coupled if the dependency between the modules is based on the fact that they communicate by passing only data.

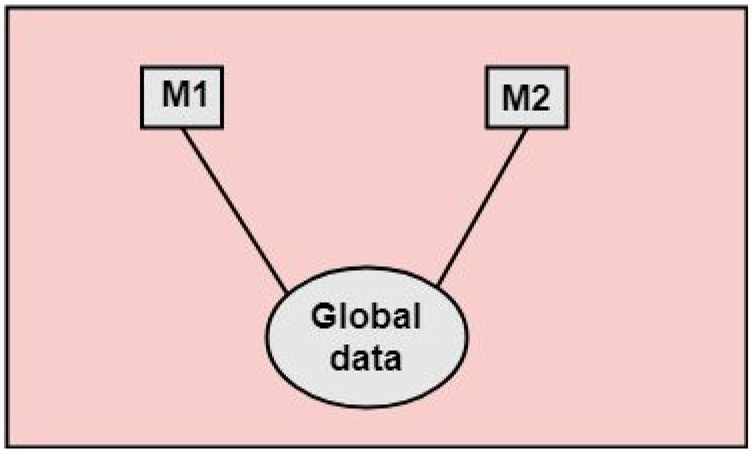


- Stamp Coupling
    - ❑ When multiple modules share a common data structure and work on different parts of it, it is called stamp coupling.
    - ❑ Complete data structure is passed from one module to another. Therefore it involves tramp data.

- Control Coupling
    - ❑ Two modules are said to be control coupling if the modules communicate by passing control information.
    - ❑ This is good when parameters allow factoring and reuse of functionality.
    - ❑ But this is bad when parameters indicate completely different behavior
      Examples : sort function that takes comparison function as an argument

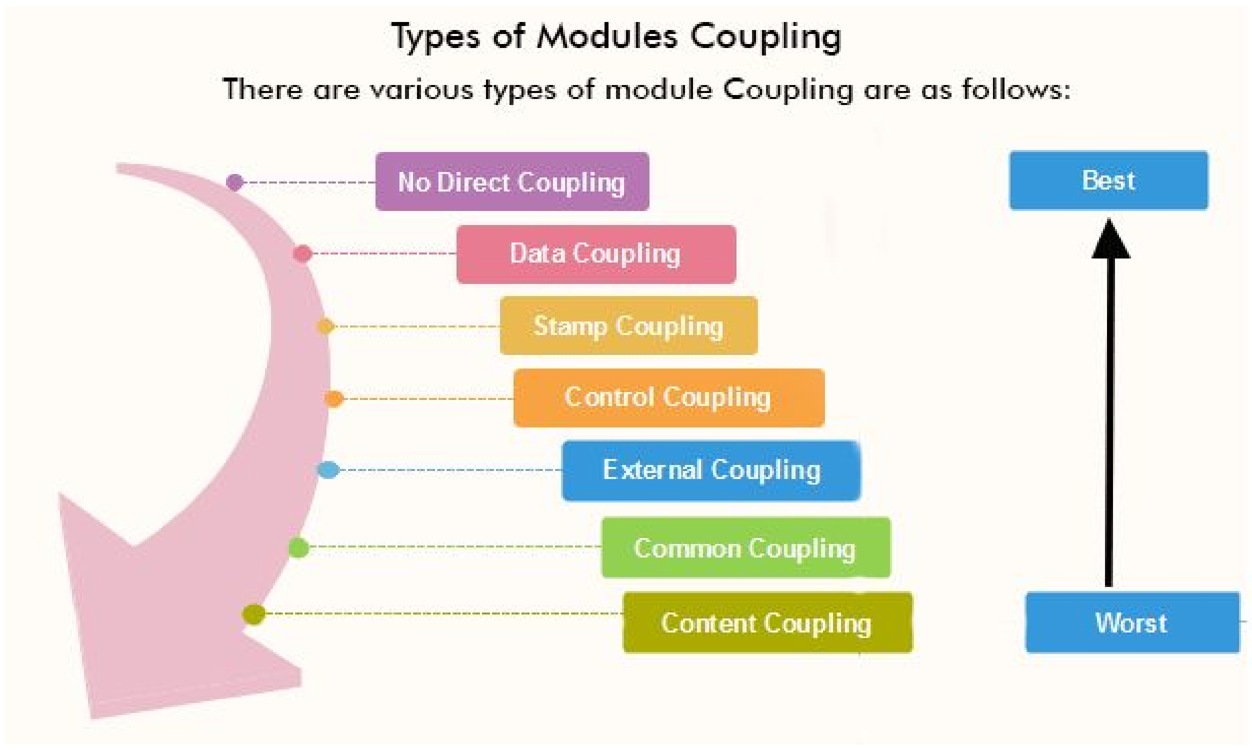
- External Coupling
    - ❑ Arises when an external imposed data format and communication protocol are shared by two modules
      Examples : protocol, external file, device format

- Common Coupling
    - ❑ Common coupling which is also known as Global coupling, occurs when two modules share the same global data. Changing the shared resource might imply changing all the modules using it.
    - ❑ Disadvantages : difficulty in reusing modules, reduced ability to control data accesses and reduced maintainability



- Content Coupling
  - ❑ When a module can directly access or modify or refer to the content of another module, it is called content level coupling.
  - ❑ This coupling type is the worst and should be avoided

- Message Coupling
  - ❑ Achieved by the state decentralization. This is the loosest type of coupling, in which the component communication is performed through message passing.



Cohesion

Cohesion is a measure of the degree to which the elements of the module are functionally related. It is the degree to which all elements directed towards performing a single task are contained in the component.

Cohesion is an ordinal type of measurement and is generally described as "high cohesion" or "low cohesion." A good software design will have high cohesion. In highly cohesive systems, functionality is strongly related.

<u>Types of cohesion</u>

- Functional cohesion
  - ❏ Exist if the different elements of a module, cooperate to achieve a single function
  - ❏ A functional cohesion performs the task and functions
  - ❏ It is an ideal situation

- Sequential Cohesion
  - ❏ When parts of a module are grouped because the output from one part is the input to another part
  - ❏ An element outputs some data that becomes the input for other elements, i.e., data flow between the parts. It occurs naturally in functional programming languages

- Communicational Cohesion
  - ❏ Two elements operate on the same input data or contribute towards the same output data
  - ❏ Example- update record in the database and send it to the printer

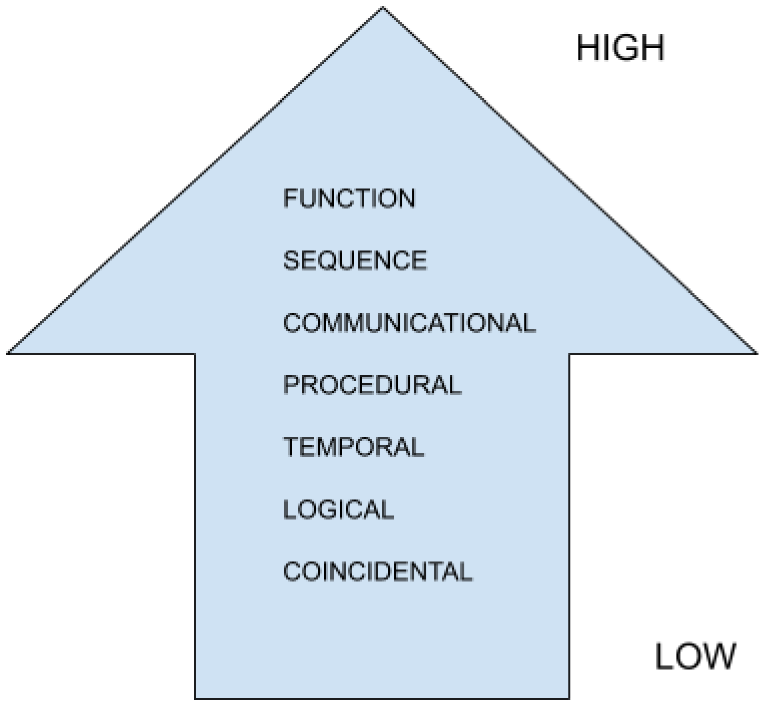
- Procedural Cohesion
  - ❏ A module is said to be procedural cohesion if the set of purpose of the module are all parts of a procedure in which particular sequence of steps has to be carried out for achieving a goal
  - ❏ Elements of procedural cohesion ensure the order of execution
  - ❏ Actions are still weakly connected and unlikely to be reusable
  - ❏ Example- calculate student GPA, print student record, calculate cumulative GPA, print cumulative GPA

- Temporal Cohesion
  - ❏ The elements are related by their timing involved

- ❏ A module connected with temporal cohesion all the tasks must be executed in the same time-span. This cohesion contains the code for initializing all the parts of the system. Lots of different activities occur, all at init time

- Logical Cohesion
  - ❏ A module is said to be logically cohesive if all the elements of the module perform a similar operation
  - ❏ The elements are logically related and not functionally
  - ❏ Example - A component reads inputs from tape, disk, and network. All the code for these functions is in the same component. Operations are related, but the functions are significantly different

- Coincidental Cohesion
  - ❏ The elements are not related(unrelated)
  - ❏ A module is said to have coincidental cohesion if it performs a set of tasks that are associated with each other very loosely
  - ❏ The elements have no conceptual relationship other than location in source code
  - ❏ It is accidental and the worst form of cohesion
  - ❏ Example - print next line and reverse the characters of a string in a single component


HIGH
FUNCTION
SEQUENCE
COMMUNICATIONAL
PROCEDURAL
TEMPORAL
LOGICAL
COINCIDENTAL
LOW