FACULTY OF ENGINEERING & TECHNOLOGY

Brajesh Mishra
Assistant Professor
Department of Computer Science & Engineering

Topics Covered

Types of coupling

Cohesion

Types of Cohesion

Types of coupling

Object-oriented programming

Subclass coupling

Describes the relationship between a child and its parent. The child is connected to its parent, but the parent is not connected to the child.

Temporal coupling

It is when two actions are bundled together into one module just because they happen to occur at the same time.In recent work various other coupling concepts have been investigated and used as indicators for different modularization principles used in practice.

Dynamic coupling

The goal of this type of coupling is to provide a run-time evaluation of a software system. It has been argued that static coupling metrics lose precision when dealing with an intensive use of dynamic binding or inheritance. In the attempt to solve this issue, dynamic coupling measures have been taken into account.

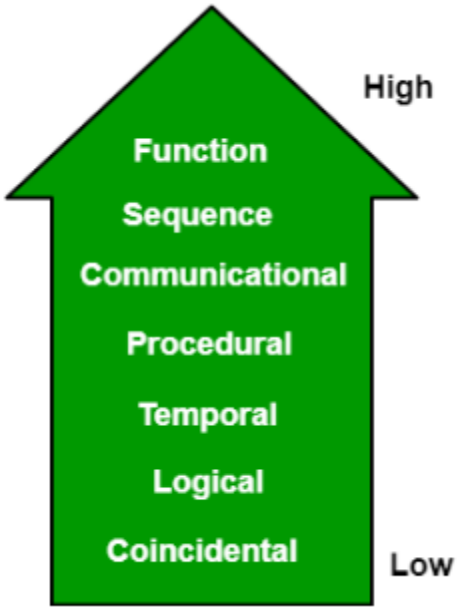
Semantic coupling

This kind of coupling considers the conceptual similarities between software entities using, for example, comments and identifiers and relying on techniques such as Latent Semantic Indexing (LSI).

Logical coupling

Logical coupling (or evolutionary coupling or change coupling) exploits the release history of a software system to find change patterns among modules or classes: e.g., entities that are likely to be changed together or sequences of changes (a change in a class A is always followed by a change in a class B).

- 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. Basically, cohesion is the internal glue that keeps the module together. A good software design will have high cohesion.



Types of Cohesion

Functional Cohesion: Every essential element for a single computation is contained in the component. A functional cohesion performs the task and functions. It is an ideal situation.
Sequential Cohesion: An element outputs some data that becomes the input for other element, 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 int the database and send it to the printer.
Procedural Cohesion: Elements of procedural cohesion ensure the order of execution. Actions are still weakly connected and unlikely to be reusable. Ex- 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.