| | |
|---|---|
| This is when a computer program causes a procedure (subroutine) to execute in a different address space (commonly on another computer on a shared network), which is coded as if it were a normal (local) procedure call | Something that is difficult or impossible to solve because of incomplete, contradictory, and changing requirements that are often difficult to recognize. It cannot be fixed, and there is no single solution to the problem; "wicked" here means resistance to resolution, rather than evil. It really has no determinable stopping point |
| In a sequential experiment they are mechanisms or procedures aimed at deciding when that experiment should continue or end, on the basis of the present position or past events and which will almost always lead to a decision to stop at some point | It represents calling relationships between subroutines in a computer programme. Each node represents a procedure and each edge (for example **f, g**) indicates that procedure **f** calls procedure **g.** Thus, a cycle in the graph indicates recursive procedure calls. |
| A software design principle where certain aspects of a programme or module are inaccessible to clients, to prevent extensive modification to the clients, and avoid potential malfunctions | This is the property of software that measures how well software is decomposed into smaller pieces with standardized interfaces. The better this feature is designed, the stronger the division of a software into modules |
| This is the measure of the degree of interdependence between the modules. A good software will have a low degree of this feature, meaning that modules should be more autonomus | It a measure of the degree to which the elements of the module are functionally related. It is basically the internal glue that keeps the module together. A good software design will have high degree of this feature |
| The generalizing physical, details or fetaures in the study of objects or systems to focus attention on details of greater importance; it is similar in nature to the process of generalization. | This is the process of dividing a software system into multiple independent modules that work independently. This makes a software's system easy to understand, easy to maintain, and with reusable elements |

**PROCEDURE CALL:** This is when a computer program causes a procedure (subroutine) to execute in a different address space (commonly on another computer on a shared network), which is coded as if it were a normal (local) procedure call.

**WICKED PROBLEM:** Something that is difficult or impossible to solve because of incomplete, contradictory, and changing requirements that are often difficult to recognize. It cannot be fixed, and there is no single solution to the problem; "wicked" here means resistance to resolution, rather than evil. It really has no determinable stopping point

**STOPPING RULES:** In a sequential experiment they are mechanisms or procedures aimed at deciding when that experiment should continue or end, on the basis of the present position or past events and which will almost always lead to a decision to stop at some point

**CALL GRAPH** it represents calling relationships between subroutines in a computer programme. Each node represents a procedure and each edge (for example f, g) indicates that procedure f calls procedure g. Thus, a cycle in the graph indicates recursive procedure calls.

**INFORMATION HIDING:** a software design principle where certain aspects of a programme or module are inaccessible to clients, to prevent extensive modification to the clients, and avoid potential malfunctions

**MODULARITY:** This is the property of software that measures how well software is decomposed into smaller pieces with standardized interfaces. The better this feature is designed, the stronger the division of a software into modules.

**COHESION:** It a measure of the degree to which the elements of the module are functionally related. It is basically the internal glue that keeps the module together. A good software design will have high degree of this feature

**COUPLING** This is the measure of the degree of interdependence between the modules. A good software will have a low degree of this feature, meaning that modules should be more autonomus

**MODULARIZATION:** This is the process of dividing a software system into multiple independent modules that work independently. This makes a software's system easy to understand, easy to maintain, and with reusable elements

**ABSTRACTION** The generalizing physical, details or fetaures in the study of objects or systems to focus attention on details of greater importance; it is similar in nature to the process of generalization.