

Subtypes Description

SubType	Macro Category	Taxonomy Type	Description
Naming	Evolvability	Textual	Naming defects usually occur when the name of a software element is meaningless or does not respect any naming policy constrain.
Comments	Evolvability	Textual	Comment defects occur when comments need to be updated due to, for instance, new information needed to describe the code after a modification.
Debug Info	Evolvability	Textual	Debug defects occur when logging messages or debug messages need to be update as the information they contain is not relevant anymore.
Other Textual	Evolvability	Textual	Other Textual defects includes all textual defects impossible to place in another existing textual subtype.
Element Type	Evolvability	Supported by Language	Element Type defects occur when the type of software element is incorrect (only cases not causing bugs). For instance, a method returning the wrong variable type.
Visibility	Evolvability	Supported by Language	Visibility defects occur when a software element has the wrong scope, like public variables that need to be private or vice versa.
Void Parameter	Evolvability	Supported by Language	Void Parameter defects occur as empty variable or brackets are used to represent void elements. In other words, when an incorrect use of the keyword “void” occurs.
Element Reference	Evolvability	Supported by Language	Element Reference defects occur when incomplete names are used to refer to a software element. A typical example is the misuse of “this” keyword.
Bracket Usage	Evolvability	Visual Representation	Bracket Usage defects occur as the use of brackets in the code is incorrect.
Indentation	Evolvability	Visual Representation	Indentation defects occur when the indentation of the code is wrong and makes the code difficult to read.
BlankLine Usage	Evolvability	Visual Representation	Blank Line usage defects occur when there are too many or too few blank line in the code formatting.
Long Line	Evolvability	Visual Representation	Long Line defects occur when a long line need to be split in separate lines to grant code readability.
Space Usage	Evolvability	Visual Representation	Space Usage defects occur as the code contains too many or too few blank space characters.
Grouping	Evolvability	Visual Representation	Grouping defects occur when software elements of the same type need to be grouped in the same packages, modules or classes or software elements of different type need to be splitted in separated packages, modules or classes.

SubType	Macro Category	Taxonomy Type	Description
Move Functionality	Evolvability	Organization	Move Functionality defects occur when a function, a part of a function, a module or a class need to be moved to a different part of the code or system.
Long Sub-Routine	Evolvability	Organization	Long Sub-Routine defects occur as complex loops or functions need to be split because of its excessive length or functionality
Dead Code	Evolvability	Organization	Dead Code defects occur when parts of code need to be removed since they are never used.
Duplication	Evolvability	Organization	Duplication defects occur when the code contains duplicated parts that need to be optimized.
Complex Code	Evolvability	Organization	Complex Code defects occur as the code structure is excessively complex.
Statement Issue	Evolvability	Organization	Statement Issue defects occur when statements in the code need to be combined, splitted, or reorganised.
Consistency	Evolvability	Organization	Consistency defects occur as similar software elements have different behaviours, causing the need to modify them to make their workflow similar. For instance, similar tasks should be implemented similarly in similar classes.
Other Organization	Evolvability	Organization	Other Organization defects groups all organization defects that cannot be assigned to any another organization subtype.
Semantic Duplication	Evolvability	Solution Approach	Semantic Duplication defects occur when different part of code have the same goal but they are different at code level.
Semantic Dead Code	Evolvability	Solution Approach	Semantic Dead Code defects occur as parts of code are executed but they do not serve any meaningful purpose.
Change Function	Evolvability	Solution Approach	Change Function defects occur when there is the need to update old or deprecated functions in the code.
Standard Method	Evolvability	Solution Approach	Standard Method defects occur when parts of code need to be replaced by standardized way of working. For instance, substituting magic number with predefined constants.
New Functionality	Evolvability	Solution Approach	New Functionality defects occur when the implementation of a new functionality is required to make the software more evolvable.
Other Solution Approach	Evolvability	Solution Approach	Other solution Approach defects include solution approach defects that cannot be placed in another solution approach subtype.
Variable Initialization	Functional	Resource	Variable Initialization defects occur when a variable needs to be initialized before its use in order to avoid run-time failures.
Memory Management	Functional	Resource	Memory Management defects occur when the system memory is wrongly handled.

SubType	Macro Category	Taxonomy Type	Description
Data & Resource Manipulation	Functional	Resource	Data & Resource Manipulation defects occur when data need to be manipulated in a certain way to avoid problem related to resources, like closing buffers or threads management.
Check Function	Functional	Check	Check Function defects occur when the return value of a function needs to be checked prior its usage.
Check Variable	Functional	Check	Check Variable defects occur when a variable needs to be checked before its usage.
Check User Input	Functional	Check	Check User Input defects occur when the user input needs to be controlled before its usage.
Function Call	Functional	Interface	Function Call defects occur when a function call is missing or incorrect.
Parameter	Functional	Interface	Parameter defects occur when a function call has wrong or missing parameters.
Compare	Functional	Logic	Compare defects occur when there is a mistake in a comparison statement.
Compute	Functional	Logic	Compute defects occur when a computation produces incorrect results.
Wrong Location	Functional	Logic	Wrong Location defects occur when a correct operation have been executed in the wrong place inside the code: it should have been performed earlier or later in the code flow.
Algorithm/Performance	Functional	Logic	Algorithm/Performance defects occur when the computation doesn't meet the efficiency requirements. In other words, when the performance of an algorithm must be improved.
Other Logic	Functional	Logic	Other Logic defects incorporates logic defects that cannot be placed in other logic defects subtypes.
Completeness	Functional	Larger	Completeness defects occur when a feature is not completely implemented, producing runtime failure or incorrect results.
GUI	Functional	Larger	GUI defects occur as there are inconsistencies in the user interfaced leading to errors.
Check Outside the Code	Functional	Larger	Check Outside the Code defects occur when a modification requires to check parts of code that were not among the ones under review.