Taxonomy *Integration of existing taxonomies*

This taxonomy was accessible to our survey participants for evaluating its completeness. It contained merged and altered taxonomy categories obtained from merging our existing taxonomy with the one from Beller *et al.*

Artifact	Activity	Category	Topic	Detailed Change
Production & Test Code	**Maintainability / Perfective Maintenance** (Modification of a software product after delivery to improve performance or maintainability)	**Documentation**	**Textual Documentation** Issues concerning the documentation through textual representation, such as naming of classes, method, variables. This also includes license headers, typos in either line comments or javadoc	**Naming** Problems relating to software element (=methods, classes, variables, etc) names that do not conform to the naming policy of the project
				Comments Explanations of complex code fragments, classes, methods. Issues include wrongly placed comments, missing comments, missing or wrong Javadoc etc.
				License Header Issues regarding missing or wrong license headers inside source-files
				Other
			Language Supported Documentation Documentation through statements/elements that the programming language offers (e.g., java public modifier to document that it is accessible from the outside)	**Immutability** Not declaring variable to be immutable when it should have been or declaring it immutable when it should have not been
				Visibility (Modifiers) Software element (e.g. method, variable, class) has too much or too restricted visibility
		Style		**Brackets & Braces** e.g., single statement after a conditional branch
				Indentation consistent indentation of the code
				Blank Lines excess of blank lines or too few blank lines or wrong split of lines
				Long Lines code statement too long, over a specific amount of characters
				Whitespace Usage usages of blank spaces in the code
				Grouping grouping of methods with related functionality or adding class variables at the beginning of the class
				Commented out code remove code that is commented out (also TODO and FIXME)
		Structure	**Solution Approach** Solution approach defects require an alternative implementation method. For example, replacing the program’s array data structure with a vector and knowing the existence of prebuilt functionality that could be used instead of a self-programmed implementation would be considered a solution approach defect. Therefore, solution approach defects are not about reorganizing existing code but rethinking the current solution and implementing it in a different way.	**Semantic Duplication** Code structures that have a similar intention but are implemented syntactically different
				Semantic Dead Code Code fragments that are executed, but they do not serve any meaningful purpose and/or have no effect on the result
				Change Function Change function call to another function because it uses old or deprecated functions
				Standard Coding Conventions Use exceptions for error messaging instead of return values, use predefined constants instead of magic numbers etc.
				New Functionality new functionality to ensure evolvability, e.g., create new classes, methods to make code more maintainable
				Testing Issues regarding test coverage, wrong tests, additional tests etc.
				Other
			Organization Defects that can be fixed by applying structural modifications to the software. Moving a piece of functionality from module A to module B is a good example of this.	**Imports** Issues with wrong or missing or unused import statements
				Move Functionality move functions, part of functions, or other functional elements to a different class, file, or module
				Long Sub Routine split long and comples functions into multiple functions
				Dead Code remove code that is never reached and executed
				Duplication / Redundant Code remove duplicate code or code that is not used
				Complex Code / Simplification restructure or rewrite implementation to make it more understandable
				Statement Issue splitting, combining or otherwise reorganizing a statement inside a function
				Consistency Means the need to keep code consistent in a sense that similar code elements operate in a similar fashion and are more or less symmetrical. For example, similar tasks in similar classes should have similar implementations
				Other
	Functionality / Corrective Maintenance (Reactive modification of a software product performed after delivery to correct discovered problems.)	**Interface** Communication with a different part of the system		**Function Call** call to another part of system or libary is incorrect or missing
				Parameter function call or other interaction has incorrect or missing parameters
		Logic		**Compare** mistake in a comparison statement
				Compute computations produce incorrect results
				Wrong Location correct operation is performed, but it is done too soon or too late
				Algorithm/Performance inefficient algorithm is used
				Other
		Resource		**Variable Initialization** Variables are left uninitialized prior to use. Uninitialized variables may contain any value and using such variable for comparison or calculation produces arbitrary results.
				Memory Management Mistake is made in handling the system memory.
				Data & Resource Manipulation Defects related to manipulating or releasing data or other resources.
		Check		**Check Function** when a function is called there is also a need to check that the value returned is valid and that no error occurred
				Check Variable there is a need to check variable
				Check User Input the need to validate user input
		Larger Defects		**Completeness** partially implemented feature
				GUI Defects in the user interface code relating to the consistency of the user-interface, and to the options made possible to the user in each situation.
				Check outside code Defects that required that part of the application code that was not under review to be checked, as it was likely to contain incorrect code based on the current review.

Other Changes (Config files,Scripts, CI/CD, README etc.)	**Commit Message**
	CI / CD configurations
	ASAT configurations
	Language or Framework specific
	Scripts
	README
	HTML
	VCS
	External Software Documentation
	Runtime Configurations docker-configs, ansible playbooks, deployment configs etc.