TABLE V
ASSOCIATIONS IDENTIFIED IN THE LITERATURE ABOUT CI AND SOFTWARE QUALITY VARIABLES.

Association	Rationale
$CI \rightarrow BugResolution$	Projects presents more resolved issues and bugs after adoption of CI [28].
$CI \rightarrow ResolutionTime$	CI is related to an increasing in the number of issues closed by period, helping to spend less time debugging and more time adding features [29], [30].
$CI \rightarrow BugReport$	CI teams discover more bugs than no-CI teams, and CI projects present fewer defects than no-CI projects [26], [27].
$CI \rightarrow Transparency$	CI is associated with a transparency increase, facilitating collaboration [34].
$CI \rightarrow Communication$	The general discussion, the number of line-level review comments, and change-inducing review comments tend to decrease after CI adoption without affecting pull request activity [35].
$CI \rightarrow Overconfidence$	CI developers are reported as suffering from a false sense of confidence (when blindly trusting the tests) [31], [32].
$CI \rightarrow TechnicalChallenges$	Configuring the build environment, the tools, and practices impose challenges for CI teams [31], [33].

TABLE VI
ASSOCIATIONS IDENTIFIED IN THE LITERATURE ABOUT BUG REPORTS.

Association	Rationale
$LackOfKnowledge \rightarrow BugReport$	Insufficient domain and linguistic knowledge are presented as possible human root causes for software defects [36].
$LackTechKnowledge \rightarrow BugReport$	Insufficient programming and strategy knowledge and failure to catch the specific feature of the problems are mapped as possible human root causes for software defects [36].
$RequiremProblem \rightarrow BugReport$	Requirement management problems and a misunderstanding of requirements and design specifications are reported as possible human causes of software defects [36].
$Overconfidence \rightarrow BugReport$	Overconfidence and confirmation bias contributes to evaluation errors and software defects [36].
$Inattention \rightarrow BugReport$	Interruptions and other kinds of inattention are reported as possible human causes of software defects [36].
$Communication \rightarrow BugReport$	Communication problems lead to expression and comprehension errors [36].
$ConfigManagement \rightarrow BugReport$	Configuration management problems lead to process errors [36].
$Tools \rightarrow BugReport$	Tools problems like compiler induced defects are possible root causes of software defects [36].
$CodeSmells \rightarrow BugReport$	Code smells on the occcurrence of bugs [37].
$NumberOfForks \rightarrow BugReport$	The number of forks has an association with an increase in bug reports [26].
$ProjAge \rightarrow BugReport$	Project age has a significant negative effect on the count of bugs reported by core developers [26].
$ProjPopularity \rightarrow BugReport$	Project's popularity has a significant negative effect on the count of bugs reported by core developers [26].
$QuantIssues \rightarrow BugReport$	The number of non-bug issue reports has a significant and positive effect on the response [26].
$TestsVolume \rightarrow BugReport$	The size of test files has a negative effect on bug reports [26].

TABLE VII
ASSOCIATIONS IDENTIFIED IN THE LITERATURE ABOUT BUG RESOLUTION.

Association	Rationale
$InternalQuality \rightarrow BugResolution$	Elements of $InternalQuality$, such as maintainability, analysability, changeability, stability, testability, project volume, duplication, unit size, unit complexity, and module coupling, present significan correlation with defect resolution efficiency [36].
$Communication \rightarrow BugResolution$	Human and data elements such as comments, severity, product, component, among others, can improve the performance of bug resolution [39].
$IssuePriority \rightarrow BugResolution$	Priority and severity are non-textual factors of a bug report that enhance the capability of bug resolution [39].

TABLE VIII

ASSOCIATIONS IDENTIFIED IN THE LITERATURE ABOUT THE RESOLUTION TIME.

Association	**Rationale**
$IssueType \rightarrow ResolutionTime$	Issue fixing times are different for different issue types [40], [41], [43], [44].
$Communication \rightarrow ResolutionTime$	The number of comments and the max length of all comments in the bug reports impact the resolution time. Bugs with little discussion tend to be resolved quickly [42], [43].
$IssuePriority \rightarrow ResolutionTime$	The severity of a bug report influences the delay before fixing it. As high the severity level, the fewer the delay [43].
$CommitSize \rightarrow ResolutionTime$	The size of code churn (number of methods) impacts the delay before fixing a bug report [43].
$OperateSystem \rightarrow ResolutionTime$	The median delay before fixing a bug found on Linux is shorter than other OS [43].
$IssueDescription \rightarrow ResolutionTime$	Increasing the literal length of the bug report description can increase delay until the team checks it as resolved [43].

TABLE IX

IINTERNAL ASSOCIATIONS CATALOGED AMONG THE LITERATURE REGARDING THE DISCOVERED VARIABLES.

Association	**Rationale**
$IssueType \rightarrow CommitSize$	The issue type is associated with the size of the code churn [45].
$IssueType \rightarrow Engagement$	Developers tend to spend more effort engaging with one another regarding new features and software extensions than in defects [41].
$IssueType \rightarrow InfoSharing$	Developers tend to share more information on defects and enhancements than support tasks [41].
$IssueType \rightarrow Communication$	A higher number of comments is associated with enhancements and defects [41].
$IssueType \rightarrow DifficultyLevel$	There is an association between the difficulty of a change and its type [44].
$Stability \rightarrow TechnicalChallenges$	The maturity of the tools, infrastructure, and CI activities imposes challenges to practitioners [33].

TABLE X

CI ASSOCIATIONS CATALOGED AMONG THE LITERATURE FROM THE PERSPECTIVE OF TEST PRACTICES.

Association	**Rationale**
$AutomatedTests \rightarrow BugReport$	Automated tests are related to improved product quality in terms of fewer defects in the software [46].
$AutomatedTests \rightarrow CodeCoverage$	Automated tests are related to high coverage of code [46].
$AutomatedTests \rightarrow WorkTime$	Automated tests are related to reduced testing time [46].
$AutomatedTests \rightarrow Confidence$	Automated tests are related to increased confidence in the quality of the system [46].
$AutomatedTests \rightarrow HumanEffort$	Automated tests are related to the less human effort that can be redirected for other activities [46].
$AutomatedTests \rightarrow Cost$	Automated tests are related to a reduction in cost [46].
$AutomatedTests \rightarrow BugDetection$	Automated tests are related to increased fault detection [46].
$AutomatedTests \rightarrow TechnicalChallenges$	Automated tests require different skills to implement them effectively [47].
$LackTechKnowledge \rightarrow AutomatedTests$	The skills level of testers could be a hindrance to test automation [47].
$Stability \rightarrow TechnicalChallenges$	The stability and maturity of the software under test affect the maintenance effort of tests [47].
$Design \rightarrow TestReusability$	Designing tests with maintenance in mind, they can be repeated frequently [46].
$TestRepetition \rightarrow Reliability$	When repeating tests, they are more reliable than single executions [46].
$ProjAge \rightarrow AutomatedTests$	The number, coverage, and maturity of automated tests increase with time [48]–[51].

TABLE XI
CI ASSOCIATIONS CATALOGED AMONG THE LITERATURE FROM THE PERSPECTIVE OF BUILD PRACTICES.

Association	Rationale
$BuildHealth \rightarrow WorkTime$	Broken builds lead to loss of time by freezing development and tests [52].
$BuildHealth \rightarrow MergeConflicts$	Broken builds lead to work blockage, which in turn leads to merge conflicts [53].
$TeamSize \rightarrow BuildHealth$	The team size relates to build breakage. Shorter teams tend to break fewer than larger ones [52].
$MultipleWorkspace \rightarrow BuildHealth$	Maintaining multiple physical structures for multiple branches is associated with more build breakage [52].
$DeveloperRole \rightarrow BuildHealth$	There is a statistical difference in build breakage among different role groups [52].
$CommitSize \rightarrow BuildHealth$	The size of the changes is related to a higher probability of build failure [52], [54], [55].
$CommitType \rightarrow BuildHealth$	The commit type (such as features and bugs) and the contribution model (e,g., pull request and push model) are associated with build breakage [52], [54], [55].
$CommitMoment \rightarrow BuildHealth$	There is an association between the moment of contributions and the rate of build breakage [52].
$TeamDistribution \rightarrow BuildHealth$	The geographical distance of the team members is associated with the build results [52].
$Tools \rightarrow BuildHealth$	The languages and their tools are related to different build breakage rates [56].
$ExtraComplexity \rightarrow BuildHealth$	Complex builds tend to break [53].
$FlakyTests \rightarrow BuildHealth$	Flaky tests favor the occurrence of build breakage [53], [55].
$ContributorType \rightarrow BuildHealth$	Less frequent contributors tend to break builds less [55].
$TimeToFix \rightarrow Costs$	The time lost relates directly to a monetary cost [52].
$Communication \rightarrow TimeToFix$	The feedback mechanisms and information speed affect the awareness of a broken build and the time to fix it [52].
$DeveloperRole \rightarrow TimeToFix$	The developer role is associated with the time to fix a broken build [52].
$CommitType \rightarrow TimeToFix$	The characteristics of the branches and code access (e.g., isolated branches) are associated with the time to fix a broken build [52].
$IntegrationFreq \rightarrow TimeToFix$	The integration frequency in the team affects the build fixing [52].
$ProgramLanguage \rightarrow TimeToFix$	The programming language is related to the time spent to fix a broken build [56].
$ErrorUnderstand \rightarrow TimeToFix$	The understandability of the build failures directly impacts the time needed to solve them [57].
$BuildFailType \rightarrow TimeToFix$	The build failure types are associated with different difficulty levels [57].
$TestsVolume \rightarrow CommitSize$	Complex and time-consuming testing is a possible reason for large commits [53].
$ContributorType \rightarrow CommitSize$	The type of contributor (e.g., casual) relates to the build breakage rate [58].
$ContributorType \rightarrow AutomatedTests$	The contributor type is related to the number of automated tests [58].
$Extracomplexity \rightarrow ContributorType$	The complexity of the jobs is related to the type of contributor in the projects [58].
$CommitSize \rightarrow MergeConflicts$	Large commits are associated with merge conflicts [53].
$FixTools \rightarrow ErrorUnderstand$	Fix support tools improves the understandability of the build logs [57].
$BuildFailType \rightarrow ErrorUnderstand$	The build failure type is associated with different levels of understandability [57].