InsighTD Survey
Investigating causes and implications of Technical Debt

List of TD causes and their categories identified in InsighTD BR

The categories identified were:

- **Lack of knowledge:** refers to the causes related to the team's lack of knowledge to develop the project. This category grouped seven causes, such as *lack of understanding* and *lack of experience*;
- **External factors:** brings together causes that are related to issues that are external to the development team and organization. With seven causes, this category grouped items such as *discontinued component* and *customers do not know their own needs*;
- **Infrastructure:** refers to causes related to tools, technologies, and development environments. With four causes, this category groups causes such as *inadequate use of tools* and *unavailable infrastructure*;
- **Methodology:** groups causes related to processes and methodologies used in the development of the project. There were 17 causes associated with this category, which includes items like *lack of a well defined process* and *lack of code review*;
- **Organizational:** groups organization level related causes. This category groups items such as *inadequate management decision* and *lack of qualified professionals*, totaling six causes;
- **People:** groups causes directly related to members of software development teams. This category grouped eight causes like *lack of commitment* and *overload of the team*;
- **Planning and management:** groups causes related to project planning and management. In this category, we have 12 as, for example, *inadequate planning* and *poor resource allocation*;
- **Development issues:** groups issues that occur during project development. With 18 associated causes, this category has items like *poor quality* and *non-adoption of good practices*.

List of causes:

Development Issues
Non-adoption of good practices
Requirements elicitation issues
Inaccurate or complex requirement
Bad design
Sloppy code
Change in design
Complexity of the project
Change of scope
Change of requirements
Lack of quality
Poor scope definition
Problems in architecture
Non-compliance with non-functional requirements
No bug fixes
Adoption of contour solutions as definitive
Inadequate data model

Version Incompatibility
Poor choice of framework

External factors

Pressure
Customer does not know his own needs
Customer does not listen to project team
External component dependency
Discontinued component
Third party team involved in the project
Structural change in the companies involved

Infrastructure

Required infrastructure unavailable
Inadequate choice of technology / tool / platform
Problems with test environment
Inappropriate use of tools

Lack of knowledge

Lack of knowledge
Lack of experience
Lack of knowledge on development tools
Lack of domain knowledge
Lack of understanding
Lack of interest in acquiring knowledge
Lack of information

Methodology

Lack of a well-defined process
Outdated / incomplete documentation
Test not performed
Lack of refactoring
Inappropriate / poorly planned / poorly executed test
Lack of code review
Nonexistent documentation
Lack of automated testing
Lack of reuse practices
Non-compliance with policies established by management
Lack of requirements analysis
Lack of change control
Lack of IT Governance
Lack of pair programming
Lack of traceability of bugs
Lack of validation

Organizational

Lack of qualified professionals
Lack of specific team
High turnover of the team
The company does not give importance to documentation
Lack of training

Inadequate management decision

People

Lack of commitment
Team overload
Lack of team communication
Low productivity
Developers do not like to do some activities
Lack of motivation
Lack of transparency between client and development team
Non-sharing of knowledge

Planning and management

Deadline
Inappropriate planning
Not effective project management
Inaccurate time estimate
Focus on producing more at the expense of quality
Inadequate impact and risk analysis
Cost
Poor allocation of resources
Lack of perception of the importance of testing and refactoring
Manager's lack of awareness of customer needs
Concern with just back-end development
Poorly crafted SLAs