UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

Relatórios Técnicos
do Departamento de Informática Aplicada
da UNIRIO
n° 0006/2017

Methods, Techniques and Tools to Support Software Project Management in High Maturity

Cristina Teles Cerdeiral
Gleison Santos

Departamento de Informática Aplicada

UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
Av. Pasteur, 458, Urca - CEP 22290-240
RIO DE JANEIRO – BRASIL

Relatórios Técnicos do DIA/UNIRIO, No. 0006/2017 December, 2017

Methods, Techniques and Tools to Support Software Project Management in High Maturity *

Cristina Teles Cerdeiral[1] Gleison Santos[1]

[1]Depto de Informática Aplicada – Universidade Federal do Estado do Rio de Janeiro (UNIRIO)

{cristina.cerdeiral, gleison.santos}@uniriotec.br

Abstract. High maturity in software development involves statistically controlling the performance of critical subprocesses and using the predictability thus gained to manage projects with better precision and control. Maturity models such as CMMI mention statistical and other quantitative methods, techniques and tools supporting high-maturity project management, but do not provide details about their use or available types. Thus, knowledge is lacking on this area. The goal of this study is to identify methods, techniques and tools which can assist in high-maturity software project management. By conducting a systematic literature mapping, we identified 108 papers describing 153 contributions. We describe the contributions identified, classifying them by their type, their software technology maturation phase, the method by which they were evaluated, the development methods and characteristics which they support, and the process/indicator areas to which they were applied. We hope this work can help fill the knowledge gap on the statistical and other quantitative methods, techniques and tools actually being proposed, evaluated, experimented with and adopted by organizations to support quantitative high-maturity software project management.

Keywords: Quantitative project management, high maturity project management.

Resumo. A alta maturidade no desenvolvimento de software envolve controlar o desempenho dos subprocessos críticos e utilizar a previsibilidade adquirida para gerenciar os projetos com melhor precisão e controle. Modelos de maturidade como o CMMI mencionam métodos, técnicas e ferramentas estatísticas e quantitativas que apoiam a gerência de projetos na alta maturidade, porém não fornecem detalhes sobre sua utilização ou tipos disponíveis. Portanto, existe uma demanda por conhecimento na área. O objetivo deste estudo é identificar métodos, técnicas e ferramentas que possam auxiliar na gerência de projetos na alta maturidade. Através de um mapeamento sistemático da literatura, foram identificados 108 artigos descrevendo 153 contribuições. As contribuições foram descritas e classificadas por tipo, fase de maturação tecnológica, método pelo qual foram avaliadas, métodos e características de desenvolvimento que apoiam, e processos e áreas de medição nas quais foram aplicadas. Esperamos que este trabalho possa contribuir com informações sobre os métodos, técnicas e ferramentas estatísticas e quantitativas sendo propostas, avaliadas, experimentadas e adotadas pelas organizações para apoiar a gerência de projetos na alta maturidade.

Palavras-chave: Gerência quantitativa de projetos, gerência de projetos na alta maturidade.

* (Work funded by CAPES, FAPERJ (Projects E-26/010.000883/2016, E-26/010.001644/2016), and UNIRIO (Project PQ-UNIRIO 01/2016)).

Table of Contents

1 Introduction

Software development companies seek capability and maturity models such as CMMI (CMMI PRODUCT TEAM, 2010) as a way to improve their processes maturity and their products quality. To achieve high maturity levels (such as levels 4 and 5 of CMMI), software organizations must statistically control the performance of their critical subprocesses and apply the predictability thus gained to perform quantitative project management with better planning precision and monitoring control (CMMI PRODUCT TEAM, 2010).

Project management involves applying knowledge, skills, tools, and techniques to the activities of a project to meet its requirements, achieving quality and performance goals, usually regarding defects, schedule, resources and cost restrictions (PROJECT MANAGEMENT INSTITUTE, 2017)."Traditional" project management practices, found on project management bodies of knowledge, methods, standards, and maturity models, such as PMBOK – Project Management Body of Knowledge (PROJECT MANAGEMENT INSTITUTE, 2017), PRINCE2 - PRojects IN a Controlled Environment (AXELOS, 2017), ISO 21500 (ISO, 2012) and PMMM – Project Management Maturity Model (CRAWFORD, 2006) – which consist of preparing project plans and analyzing collected measures, and comparing the obtained results against the plans – are not enough to achieve the predictability needed for high maturity (FENTON et al., 2004). Instead of reactively making action plans when the values obtained do not match the planned ones, high-maturity project management avoids the problems proactively by predicting them before they occur (BHARATHI; SHASTRY; RAJ, 2012).

To achieve high maturity in software development, software organizations need to analyze their historical data and understand and control their critical subprocesses to gain knowledge about their statistical stability, performance limits, and capability, and to establish feasible organizational improvement goals for those subprocesses. Project managers must evaluate whether quality and performance project goals are in accordance with the performance, capability and improvement goals of the organization's subprocesses, and build the project process taking into consideration the performance of its subprocesses, in order to better meet quality and performance project goals. Likewise, project managers must monitor project process performance against the performance and capability of the organization's subprocesses (CMMI PRODUCT TEAM, 2010).

There are several methods, techniques and tools which can support software organizations in analyzing and controlling their critical subprocesses and in performing quantitative high-maturity project management. Maturity models such as CMMI provide some guidance in this matter, suggesting several of them. Regarding statistical techniques, the model cites process performance models, process performance baselines, statistical process control charts, regression analysis, variance analysis, prediction intervals, hypothesis testing, simulation, sensitivity analysis, and time series analysis. Regarding other quantitative techniques, the model mentions scatterplots, histograms, box and whiskers plots, Ishikawa diagrams, and Pareto analysis (CMMI PRODUCT TEAM, 2010).

However, many companies report difficulties in achieving high maturity and quantitatively managing projects. The most commonly reported difficulties are: the effort to gather, understand, analyze, and scrub data to ensure its integrity (GONÇALVES, L. et al., 2012; LEE, Dalju; BAIK; SHIN, 2009; SCHOTS et al., 2014; SHARMA, D. et al., 2016; TAKARA et al., 2007); the amount of historical data needed to achieve confidence in statistical analysis (GOU et al., 2009; LEE, Dalju; BAIK; SHIN, 2009); the need for solid correlation between organizational and project goals, critical subprocesses that support those goals, and the things being measured to provide insight into the performance of subprocesses (GROSSI; CALVO-MANZANO; SAN FELIU, 2014; LEE, Dalju; BAIK; SHIN, 2009; SHARMA, D. et al., 2016; TAKARA et al., 2007); and the need for project managers and process groups to consult specialists for appropriate guidance on statistical knowledge (CARD; DOMZALSKI; DAVIES, 2008; GONÇALVES, L. et al., 2012; GOU et al., 2009; LEE, Dalju; BAIK; SHIN, 2009; SCHOTS et al., 2014; SHARMA, D. et al., 2016; TAKARA et al., 2007).

Therefore, high-maturity project management is not a trivial task and one related problem is the absence of more detailed information about statistical and other quantitative methods, techniques and tools (e.g., control chart types and simulation algorithms available). In technical report, we use the systematic literature mapping (SLM) method (KITCHENHAM; CHARTERS, 2007; PETERSEN et al., 2008; PETERSEN; VAKKALANKA; KUZNIARZ, 2015) to identify available studies to answer research questions regarding different methods, techniques and tools which can assist in high-maturity project management. We believe this knowledge can help software process improvement initiatives to choose and use statistical and other quantitative methods, techniques and tools more appropriate to their context and needs.

The remainder of this paper is organized as follows. Section 2 describes basic principles of project management in high maturity. Section 3 describes the SLM method used in this review. Section 4 describes the execution of the review, including quality assessment of papers. Section 5 presents the analysis of the results of the review including an overview of the papers found. Section 6 describes related work. Section 7 discusses the threats to the validity of this work. Finally, Section 8 presents conclusions.

2 Project Management in High Maturity

The *first step* towards high maturity levels (Level 4 – Quantitatively Managed, and Level 5 – Optimizing of CMMI) is to identify organization's critical subprocesses which impact the achievement of the organization's goals, and statistically control their performance. Statistical process control (SPC) is a methodology for controlling process quality and performance, assisting in identifying problems, and taking actions in order to stabilize process performance (CMMI PRODUCT TEAM, 2010; FLORAC; CARLETON, 1999).

A process can have two possible types of variation. Common-cause (or chance cause) variation, characterized by a stable and consistent pattern of measured values over time, is the result of normal or inherent interactions among the people, machines, materials, environment, and methods of a process. Special-cause (or assignable cause) variation, characterized by sudden or persistent abnormal changes in one or more process components, involves events that are not part of the normal process (FLORAC; CARLETON, 1999; WHEELER, 2000).

There are several techniques which can be used to support statistical process control, with the most popular one being control charts as proposed by Shewhart (SHEWHART, 1931). These charts show data collected from process executions ordered by time, making it possible to analyze process performance, stability and capacity. A typical control chart contains a center line representing the mean value of the process attribute, and two horizontal lines: the upper control limit (UCL) and lower control limit (LCL).

Nelson proposed eight rules which have been frequently adopted to help identify special-cause variations (FLORAC; CARLETON, 1999; NELSON, 1984; WHEELER, 2000). Each special-cause variation should be analyzed to identify its causes. Action plans should then be developed to address those causes and avoid (in the case of worse performance) or ensure (in the case of better performance) the abnormal behavior. This cycle goes on until there are no special-cause variations in a considerable number of executions and the process is under control or stable. A stable process is a process with predictable performance, costs and quality and measurable change effects (CMMI PRODUCT TEAM, 2010; FLORAC; CARLETON, 1999).

The *second step* towards high maturity levels is generating performance baselines for critical subprocesses using data from past executions to manage future executions. Subprocess performance baselines can be used in project planning and monitoring in many ways, e.g. to avoid accepting or establishing unfeasible project quality and performance objectives; to decide which subprocesses will be combined into the project process in order to achieve project quality and performance objectives; to select a resource allocation having better chances to achieve project quality and performance objectives; to assess the risks in achieving project quality and performance objectives; to identify project performance issues in a quantitative manner, and to monitor action plan results. Project data alone can also be used to generate project performance baselines, which usually present less variation than subprocess performance baselines with data from different projects, allowing abnormal performance issues to be identified sooner (CMMI PRODUCT TEAM, 2010; FLORAC; CARLETON, 1999).

The *third step* towards high maturity levels is creating performance models to assist project management. A performance model allows identifying statistical correlations between different activities or project attributes and helps gaining better understanding of the processes (CMMI PRODUCT TEAM, 2010). One example is finding the correlation between time spent on requirements peer reviews and the number of defects found later, in software tests. Performance models can be used in project management to predict and estimate project outcomes, helping to develop more accurate plans to achieve project quality and performance objectives, quantitatively predict project performance issues before they occur, and predict the results of action plans.

Many different statistical and other quantitative methods, techniques and tools can be applied in these three steps to support high-maturity project management, such as the use of a specific control chart, or a specific method to generate process performance baselines, or specific statistical performance models.

3 Research Method

A systematic literature review (SLR) is a means of identifying, evaluating and interpreting the available research related to a research question, topic area, or phenomenon. The main purpose for conducting a systematic literature review is to gather evidence on which to base conclusions (KITCHENHAM; CHARTERS, 2007). A systematic literature mapping (SLM) adopts the same rigor as an SLR but its main purpose is to map the available evidence when no conclusions can be reached (PETERSEN et al., 2008; PETERSEN; VAKKALANKA; KUZNIARZ, 2015).

To conduct this SLM, we used the guidelines proposed by Kitchenham and Charters (2007) and Petersen et al. (2008; 2015). These guidelines define several steps grouped in three phases (Figure 1). It is important to note that this process is incremental, undergoing iterations and adjustments as greater understanding is gained of the topic being studied.

The following subsections detail important aspects of the research protocol related to the steps involved in SLM. The first step in SLM is identifying a need, which in our case is reinforced by the absence of any similar study providing a systematic mapping of the methods, techniques and tools supporting high-maturity project management, and the difficulties described by experience reports indicating that high-maturity project management is not a trivial task.

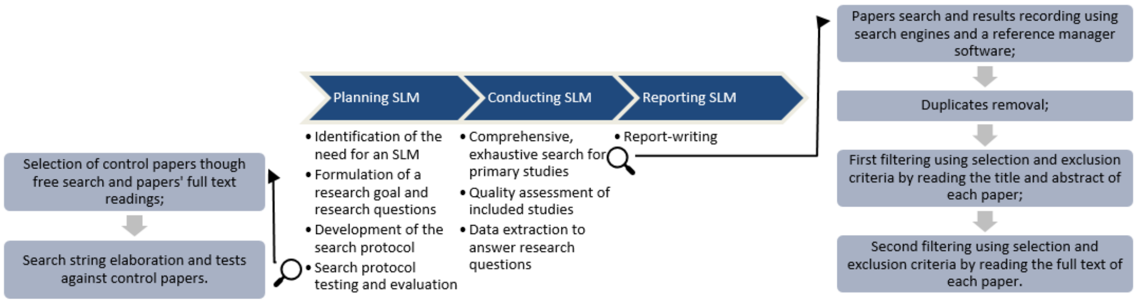

Figure 1. Steps and phases of Systematic Literature Mapping (SLM), adapted from (KITCHENHAM; CHARTERS, 2007)

3.1 Research goal and research questions

The next step in SLM is elaborating its goal to fulfill its need. The goal of the present study, stated according to the GQM paradigm (BASILI; CALDIERA; ROMBACH, 1994), is:

Analyze *quantitative project and process management improvement proposals and experiences in software engineering* ***for the purpose of*** *identifying methods, techniques and tools* ***with respect to*** *managing projects and process quantitatively* ***from the point of view of*** *software organizations* ***in the following context:*** *high maturity.*

Based on the research goal, we formulated several research questions to be answered. The primary research question can be seen in Table 1 as RQ1. To better characterize the methods, techniques and tools identified, secondary research questions were also established (RQ2 to RQ13 in Table 1).

Table 1. Primary and secondary research questions

RQ1. What are the methods, techniques and tools available which can assist in quantitative project management in high maturity context?
RQ2. What is the type of the proposed/used methods, techniques or tools?
RQ3. What existing methods, techniques or tools are being used to compose the proposed methods, techniques or tools?
RQ4. What adaptations or improvements were suggested in existing methods, techniques or tools?
RQ5. What are the expected inputs and produced outputs of the proposed methods, techniques or tools?
RQ6. What were the processes / indicators / metrics used with the proposed methods, techniques or tools?
RQ7. Do the proposed methods, techniques or tools assist in some area, domain or development method?
RQ8. In case the proposed methods, techniques or tools were developed based in historical data, what are the data sources?
RQ9. What were the evaluation techniques applied to the proposed methods, techniques or tools?
RQ10. In case of performance comparisons, what other methods, techniques or tools are compared to the proposed methods, techniques or tools?
RQ11. What are the conclusions about the proposed methods, techniques or tools?
RQ12. Were the proposed methods, techniques or tools used in ongoing projects?
RQ13. What are the observed results of applying the proposed methods, techniques or tools?

3.2 Data sources selection, search period and languages

We initially selected as possible data sources the search engines: IEEE Xplore,[1] Scopus,[2] ISI Web of Science,[3] EI Compendex,[4] and the ACM Digital Library.[5] Since many of those search engines have repeated editors and papers, we evaluated their coverage though previous studies (KITCHENHAM; CHARTERS, 2007; MATALONGA, Santiago; RODRIGUES; TRAVASSOS, 2017; OLIVEIRA et al., 2017; SILVA; COSTA VALENTIM; CONTE, 2015) and decided to keep the first four ones.

We did not manually include any conferences or journals because none of them covers the specific topics of statistical process control or quantitative high-maturity project management.

We decided on a period of fifteen years, identifying the methods, techniques and tools proposed, evaluated, experimented with and adopted during that period. We assume that, if some method, technique or tool proposed before that period had good acceptance, it would probably be mentioned in experience reports of the last fifteen years. Therefore, the search period is from 2003 to 2017.

We selected as languages Portuguese, English and Spanish, since those are the ones which the authors can understand.

3.3 Search string

To start the research, we did some exploratory readings. A fellow researcher had a group of papers selected on a previous research on statistical process control area (not necessarily project management related). The first author of this research read those papers and applied the same selection criteria and steps to select the ones that would become part of a control group (marked with ★ in Table A1 in Appendix A), used to evaluate the search string coverage.

The purpose of these exploratory readings was to become familiar with the type of information reported by the papers, in order to provide guidance on inclusion and exclusion criteria and refinement of the research questions. In addition, the keywords of the selected papers were used to help expand the search string with equivalent terms.

The defined search string can be observed in Table 2, in ISI Web of Science format. We based our search string definition on the PICOC (*Population-Intervention-Comparison-Outcome-Context*) methodology (PETTICREW; ROBERTS, 2006). Our *population* is quantitative project and process management improvement proposals and experiences in software engineering. Our *intervention* is the available methods, techniques and tools which can assist in quantitative high-maturity project management. Since these might be of several types, such as tools, processes, control charts, performance models, or frameworks, we decided not to include their types in the search string, simply mapping all types found. *Comparison* is not usually performed in SLM, since the purpose of SLM is to map the available evidence when no conclusions can be reached. Our *outcome* is the results obtained by the intervention. Since our goal is to identity interventions and their

1 http://ieeexplore.ieee.org

2 http://www.scopus.com

3 http://apps.webofknowledge.com

4 http://www.engineeringvillage.com

5 http://dl.acm.org

results, we decided not to restrict the search to a specific type of outcome, simply mapping them as well. Our context is high maturity.

Therefore, we defined our search string strongly based on the population and the context. The first part of the search string (before the *AND*) has the purpose of limiting the results to the area of Software Engineering. The second part of the search string (after the *AND*) has the purpose of limiting results to the high-maturity context, by narrowing the papers to those involving quantitative project management (since they might address the high-maturity context), quantitative process management (since they might apply to project management in the high-maturity context), high maturity levels and statistical process control (since they might describe some approach that can assist in high-maturity project management).

Whenever the search engine offered the possibility to filter data by type, we used that to narrow down the population to articles from journals and papers from conference proceedings. Whenever the search engine offered the possibility to filter data by research area we used that to narrow down the population as close as we could to Software Engineering.

Table 2. Search string in ISI Web of Science format

(("software process" OR "software development" OR "software maintenance" OR "software engineering" OR "CMMI" OR "CMM") AND ("quantitative project management" OR "quantitative process management" OR "quantitative management" OR "high maturity" OR "statistical process control" OR "statistical control" OR "statistical management" OR "control chart" OR "level 4" OR "level 5"))

We did not restrict papers to ones addressing project management explicitly. Papers which did not address project management, but which did present a method, technique or tool at the organizational level which could also assist in quantitative high-maturity project management, were selected.

3.4 Paper selection and quality criteria

Selection of papers followed the guidelines proposed by Kitchenham and Charters (2007) and Petersen et al. (2008; 2015), using the steps shown in Figure 1. Before performing the present study, the established protocol was reviewed and approved by the second author, who has more experience in SLR.

After the exhaustive search on the data sources using the search string, the papers found were recorded and analyzed. We defined selection criteria to help in identifying those papers which provide direct evidence about the research questions, and to reduce the likelihood of bias (KITCHENHAM; CHARTERS, 2007). The first filtering considered only the title and abstract of the papers. The second filtering considered the full-text reading of the papers. In both filters, papers were included if they matched one of the inclusion criteria (Table 3) or excluded if they matched certain exclusion criteria (Table 4). The selection criteria evolved during the selection process, as the understanding of the search area improved.

Table 3. Inclusion criteria

IC1. The paper addresses high-maturity project management.
IC2. The paper addresses quantitative project management which can be applied to high-maturity project management.
IC3. The paper addresses statistical process control (SPC) mentioning methods, techniques or tools which can be applied to high-maturity project management.
IC4. The paper addresses statistical methods, techniques or tools which can be applied to high-maturity project management.

Table 4. Exclusion criteria

EC1. The paper does not address high-maturity project management or present any statistical methods, techniques or tools that can be applied to high-maturity project management.
EC2. The paper does not have an abstract.
EC3. The paper is only an abstract.
EC4. The paper is not written in English, Portuguese or Spanish.
EC5. The paper is a copy or older version of an already considered paper.
EC6. The paper is not peer-reviewed (such as editorials, summaries of keynotes, tutorials).
EC7. The full text of the paper could not be accessed.
EC8. The paper is not about the area of software engineering.

To evaluate the selected papers, we defined quality criteria (Table 5), based on reviews and guidelines found in (IVARSSON; GORSCHEK, 2011; KUHRMANN et al., 2015; SHAW; SHAW, 2003). We adapted the criteria found in those studies to better classify the papers identified.

Table 5. Quality criteria

QC1. Does the paper state its goal or research goal clearly? (1. No; 2. Yes)
QC2. What is the study design? (1. Empiric; 2. Experience report; 3. Theoretical)
QC3. What were the scientific methods used to evaluate the proposed methods, techniques or tools? (1. None; 2. Example; 3. Experience; 4. Evaluation with feasibility and pilot studies; 5. Analysis.)
QC4. What research methods are used by the paper? (1. None; 2. Survey; 3. Action research; 4. Case study; 5. Experiment)
QC5. Which describes best the paper? 1. Reports an experience; 2. Reports an opinion without fundamental research; 3. Proposes a method, technique or tool; 4. Proposes and uses a method, technique or tool in academia; 5. Proposes and uses a method, technique or tool in one industry case; 6. Proposes and uses a method, technique or tool in more than one industry cases.

We believe one information that is helpful to guide the selection of which methods, techniques and tools to assist on high-maturity project management is their technological maturity. Redwine and Riddle (1985) proposed a model for the way software engineering technology evolves from research ideas to widespread practice. They defined six software technology maturation phases, as can be seen on Figure 2.

Quality criteria QC3, QC4 and QC5 were used together with RQ12 as evidence to classify the methods, techniques and tools identified in one of the maturity maturation phases. The classification rationale can be seen on Figure 2 and involved only the phases possible to identify by the information papers provide. Therefore, only the phases "Concept Formulation", "Development and Extension", "Internal Enhancement and Exploration" and "External Enhancement and Exploration" were considered.

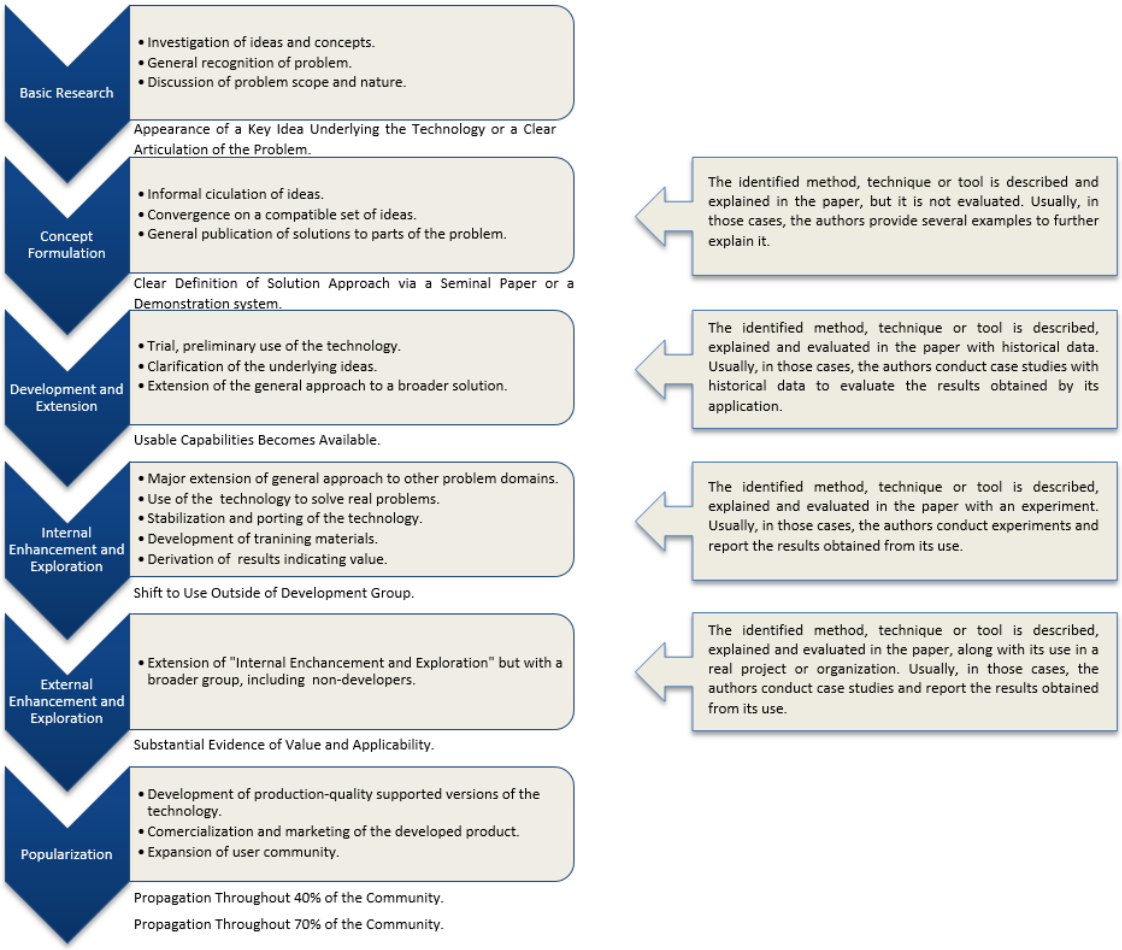

Figure 2. Software Technology Maturation Phases, adapted from (REDWINE; RIDDLE, 1985)**, and methods, techniques and tools classification rationale**

4 Execution

After the approval of the protocol, we performed the search on selected search engines and found 793 papers. All papers found were recorded using the Mendeley[6] software and analyzed in a Google Sheet[7] spreadsheet. Figure 3(a) shows the number of papers selected in each step of the selection process. Figure 3(b) shows the distribution of the 108 selected papers among the search engines, and Figure 3(c) shows the distribution of the papers for each selection filtering step for each year.

EI Compendex found 95 papers out of 108, which was the best coverage (87.96%). Scopus found 89 papers out of 108, which was the second best coverage (82.41%), but only 8 papers found by Scopus were not in the 95 ones found by EI Compendex. Together, EI Compendex and Scopus found 103 papers out of 108. ISI Web of Science found 45 papers out of 108, but only two of them were not included in the 103 papers found by EI Compendex and Scopus. IEEE Xplore found 38 papers out of 108, but only three of them were not included in the 103 papers found by EI Compendex and Scopus. The

[6] https://www.mendeley.com

[7] http://sheets.google.com

spreadsheet with all the steps and extractions performed can be found at https://goo.gl/edNsl6.

The search string has a precision of 28.20% (108 papers selected from 383 papers), thus selecting between one quarter and one third of the papers. The sensitivity of the search string is equal or less than 100%, since all control group papers were found.

The first analysis we performed was the quality analysis, with the purpose of evaluating the quality of the 108 selected papers. Table A1 in Appendix A shows the IDs, references and answers to the five quality criteria for the selected papers. We decided to sum up the answers into a quality score. Although this approach gives more impact to the criteria with more possible answers, we merely wanted to identify those papers that had extremely low scores. The result can be seen on Figure 4(a).

We found 11 papers with a score of less than 10 (marked with ♦ in Table A1). Those papers propose or report some experience with a method, technology or tool that could assist high-maturity project management, but they do not evaluate it as expected academically. It is interesting to note that seven of those 11 papers with low quality score were published between 2006 and 2003, which demonstrates that this period has more papers without appropriate evaluation. After some consideration, we decided not to exclude those 11 papers because we want to include experience reports. Instead, we decided to use QC3 and QC4 to analyze the papers to identify the method applied to evaluate each method, technology or tool, highlighting the ones lacking better evaluation.

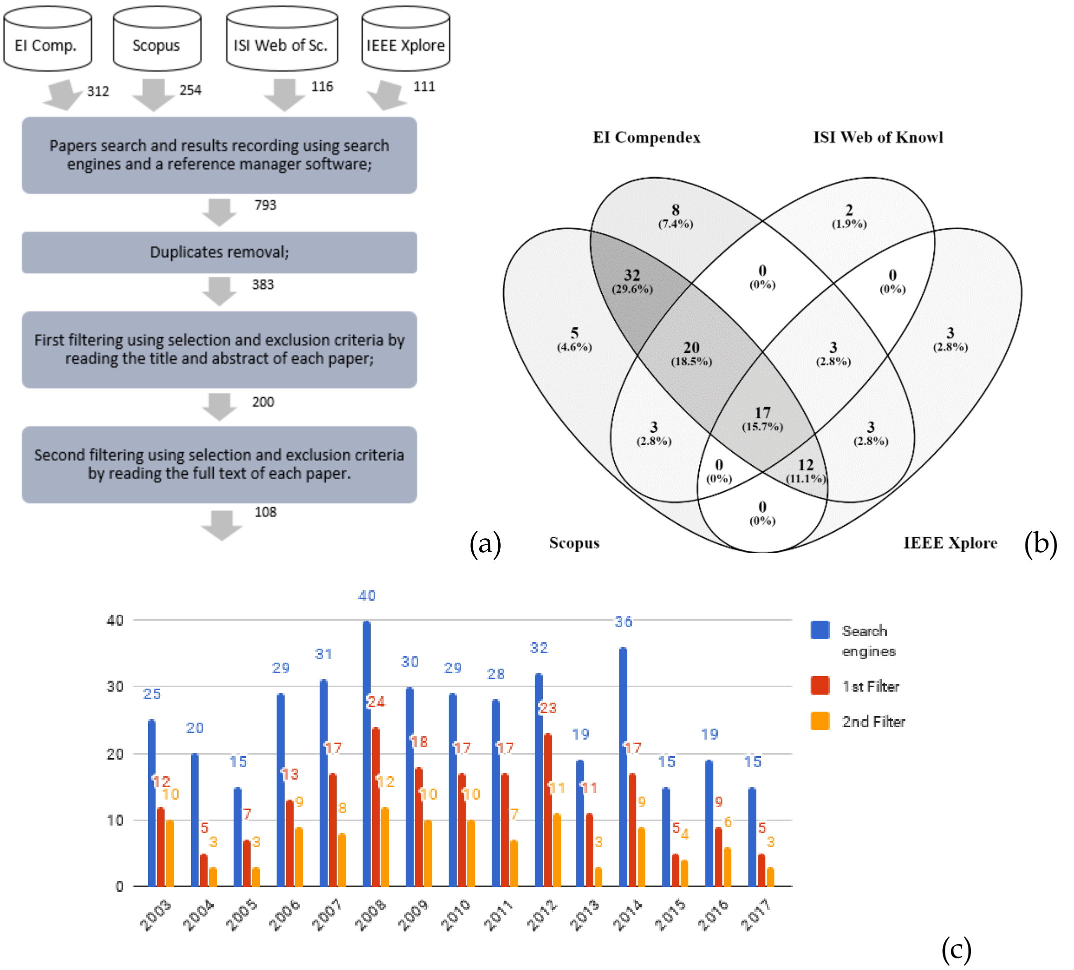

Figure 3. Distribution of papers selected

The second analysis performed was the software technology maturation phase of each identified method, technology or tool, according to the criteria on Figure 2, also shown in Table A1 in Appendix A, and on Figure 4(b). A great amount of papers provides evidence of their adoption, therefore being on the *External Enhancement and Exploration* phase, although this does not mean they were appropriately evaluated.

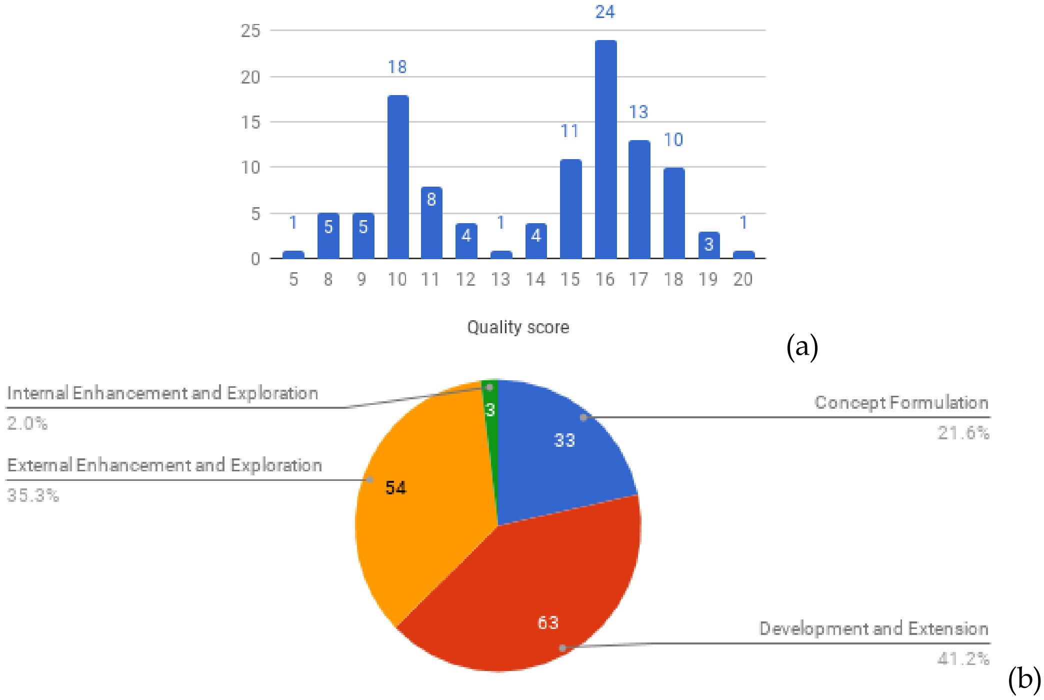

Figure 4. Quality score and software technology maturation phase of papers selected

5 Analysis of Results

The 108 selected papers were classified and analyzed against all the research questions. The first characterization is author affiliation. Authors were classified by country, and by whether they were from academia (universities or research organizations) and/or from industry (either companies developing software or consultancy companies serving them). A single paper could be classified in multiple categories. The results are shown in Figure 5(a). The USA and India are the countries with the most papers selected (31 and 19, respectively), and they have close numbers of papers from academia and industry, which suggests a strong collaboration between practitioners and researchers.

The second characterization is publication type. Figure 5(b) shows that 55 papers out of the 108 selected papers (50.9%) were published at conferences or symposiums, 2 papers (1.9%) were published at workshops, and 51 papers (51%) were published in journals. Figure 5(c) details the number of papers published per journal. The journals with more publications are IEEE Software (with 9 selected papers), followed by Cross-Talk (with 6 selected papers), and Software Quality Journal (with 5 selected papers).

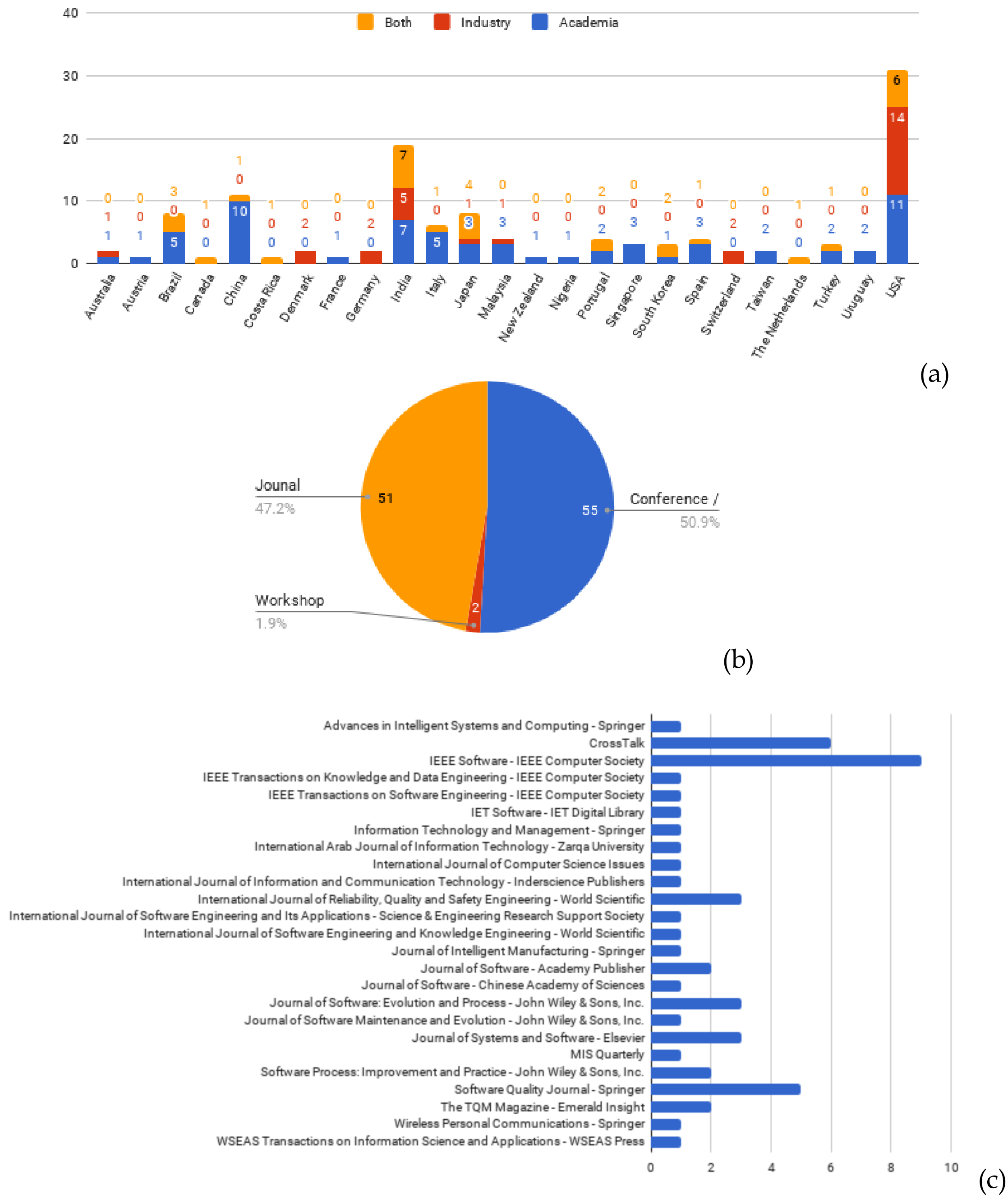

Figure 5. Author affiliation, publication type and journal distribution

RQ1. What are the available methods, techniques and tools which can assist in quantitative project management in high maturity context?

From the 108 selected papers, we identified 153 contributions (methods, techniques and tools) which can assist in quantitative high-maturity project management. We classified them by type (answering RQ2) over the years, as shown in Figure 6. The great majority of contributions identified are related to statistical performance models, control chart specifications and process performance analysis methods. Together they total 107 contributions, about two thirds of the total number identified. If we include statistical performance model building methods, since they are related to statistical performance models, the four types together total 119 contributions.

Figure 6 shows that statistical performance building methods and statistical performance models are the research topics where the level of interest has been maintained during the entire period of this search. Process performance analysis methods and tools, either automated or not, are topics showing lower interest in recent years. Control chart specifications is a topic showing interest during the entire period, with special attention in 2006. The year with most contributions identified is also 2006 (with 19 contributions), followed by 2003 and 2014 (with 15 contributions), and 2009 and 2010 (with 14 contributions).

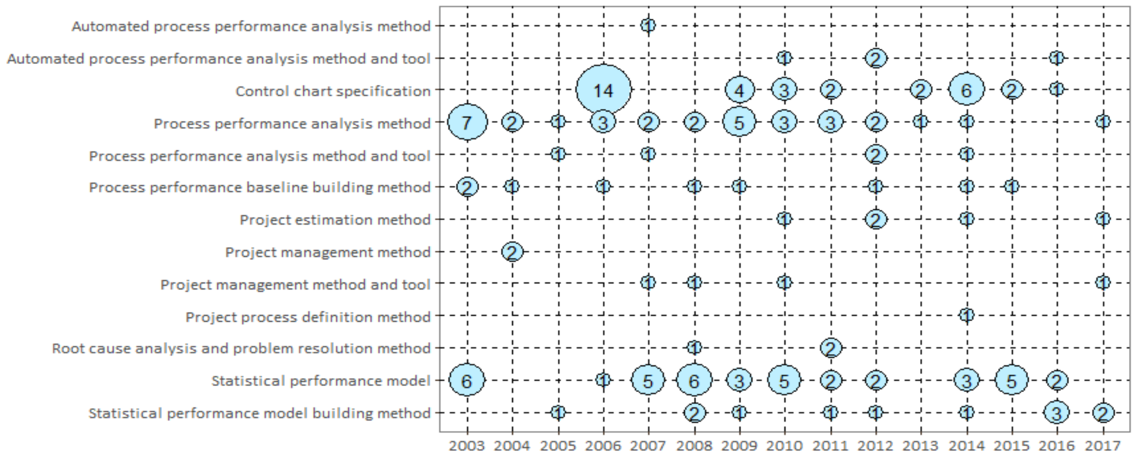

Figure 6. Types of methods, techniques and tools identified over the years

The remaining research questions are qualitative and are intended to characterize the identified methods, techniques and tools supporting quantitative high-maturity project management. Appendix B provides short versions of the answers to the remaining research questions RQ3 to RQ13 for all the contributions identified, classified by type (Tables B1 to B13. The spreadsheet with the steps and extractions performed has the extracted complete answers to all research questions for all contributions identified. We encourage anyone to check them on the spreadsheet using filtering options to better find the interesting ones.

Using the answers to research question RQ7, we classified the contributions by the development methods and characteristics which they support, as shown in Figure 7. Most of the contributions (112 out of 153) do not specify which development methods and characteristics they support. The supported development methods and characteristics which were cited most often include: maintenance/evolution and agile/iterative development.

Similarly, we used the answers to research question RQ6 to identity the process/indicator areas in which the identified methods, techniques and tools were applied to support quantitative high-maturity project management, as shown in Figure 8. In this case, we extracted the process/indicator areas used in examples or case studies, which does not necessarily mean that the contribution is targeted to support that area. In most cases, this information merely indicates which process/indicator area was used to evaluate or exemplify the application of the contribution.

We consider this information less important for the methodological types, where steps are proposed to support a task, since the method can usually be applied to any process/indicator area. We emphasize that information on types related to statistical techniques (statistical performance building methods, statistical performance models and control chart specifications), where process/indicator areas might be used to characterize and select appropriate contributions.

As we can observe in Figure 8, the majority of contributions were applied to the defects area (87 out of 153), followed by the effort area (22 out of 153). Some contributions do not detail in which process/indicator area they were applied (17 out of 153).

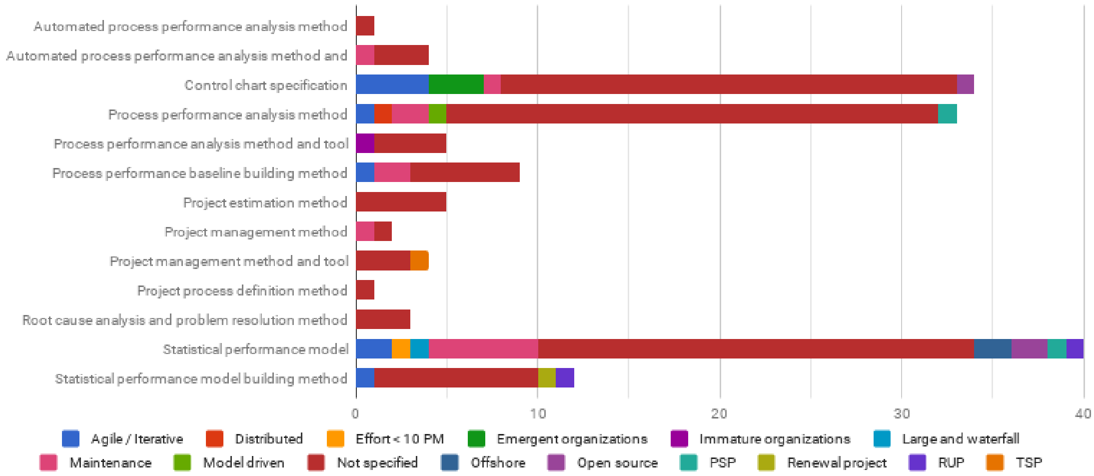

Figure 7. Types of methods, techniques and tools, classified by the development methods and characteristics which they support

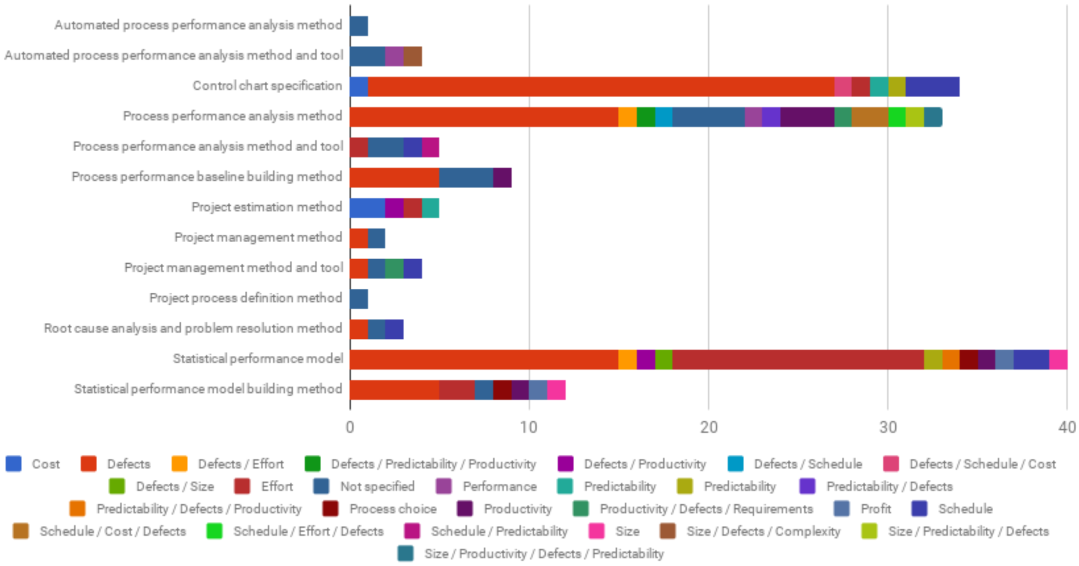

Figure 8. Types of methods, techniques and tools, classified by the process/indicator areas in which they were applied

Finally, we classified all the contributions based on their software technology maturation phase according to criteria shown in Figure 2, to assist in identifying which ones are already *adopted* by industry, versus ones which are still being *experimented with* or *evaluated*, or which have merely been *proposed*. This classification is shown in Table A1 in Appendix A. Since papers might present evidence of the contribution being adopted by industry, but without a rigorous academic evaluation of its utilization or results, we combined the answers to quality criteria QC3 and QC4 to identify the method applied to evaluate the contribution. Figure 9 shows the methods, techniques and tools which can assist in quantitative high-maturity project management, classified by their technology maturation phase, and by the method applied to evaluate them.

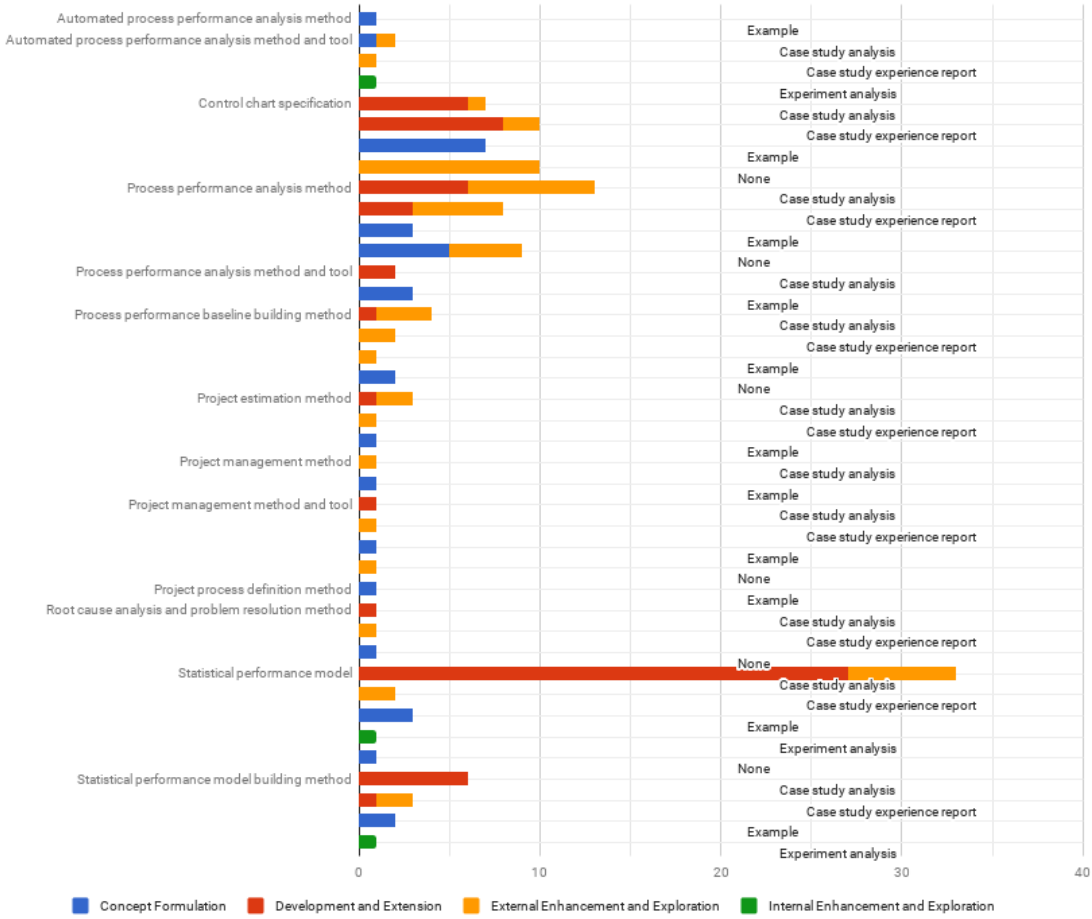

Figure 9. Types of methods, techniques and tools classified by their technology maturation phase and the method applied to evaluate them

As can be observed, most of the methods, techniques and tools identified are at the software technology maturation phase *Development and Extension* (63 out of 153), meaning they were evaluated with some data in a case study analysis or were applied to some context in a case study experience report. Figure 10 shows that most of the cases they were evaluated through a case study analysis.

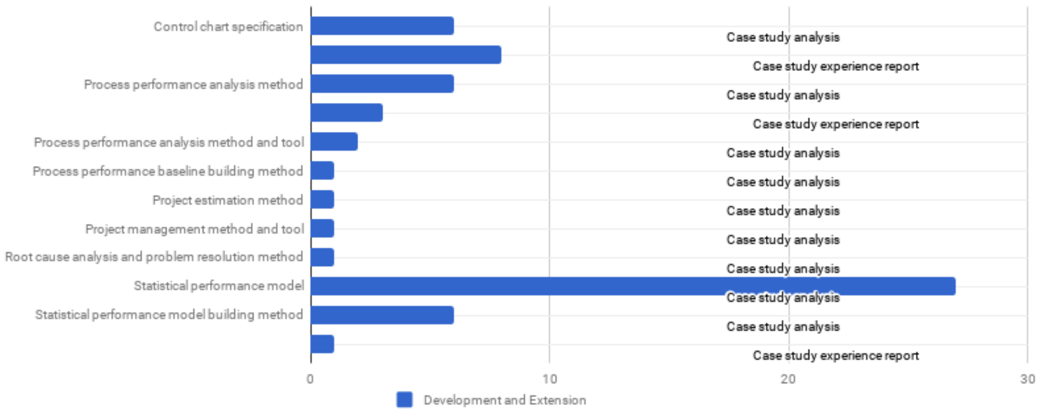

Figure 10. Types of methods, techniques and tools at the technology maturation phase *Development and Extension* and the method applied to evaluate them

The second software technology maturation phase with most contributions identified is *External Enhancement and Exploration* (54 out of 153), meaning they presented evidence of being adopted by industry. In this case, as can be seen in Figure 11, the evaluation methods vary from being adopted by an organization and analyzed in a case study analysis (21 out of 54), being adopted by an organization and described in a case study experience report (17 out of 54), to not even being evaluated at all and just being adopted by an organization and being described (15 out of 54).

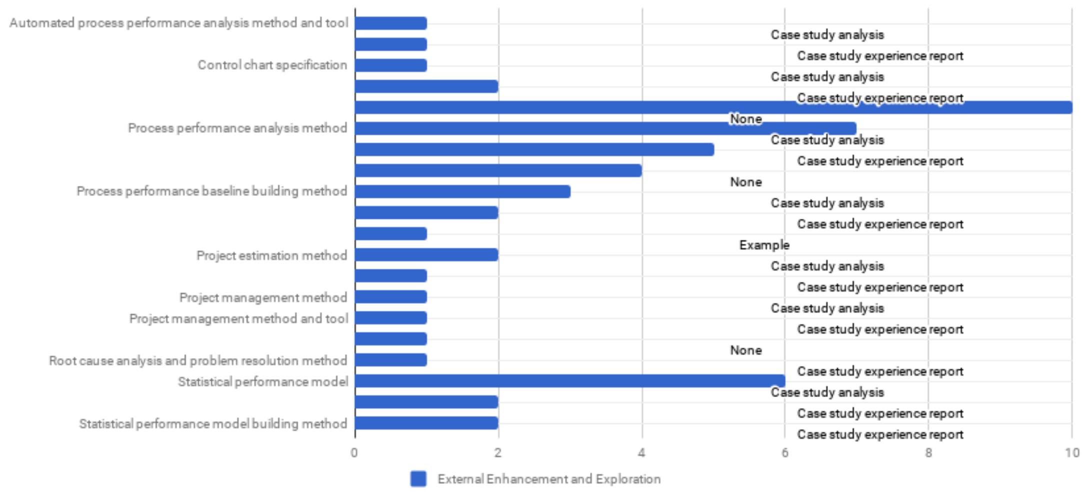

Figure 11. Types of methods, techniques and tools at the technology maturation phase *External Enhancement and Exploration* and the method applied to evaluate them

The third software technology maturation phase with most contributions identified is *Concept Formulation* (33 out of 153), meaning they were proposed or described but not evaluated academically or adopted by industry. Figure 12 shows that most of them illustrated the contributions providing examples (23 out of 33), while others just described the contributions (9 out of 33). One of them was classified as a case study analysis but it was partially evaluated on the academy, so it was not completely evaluated.

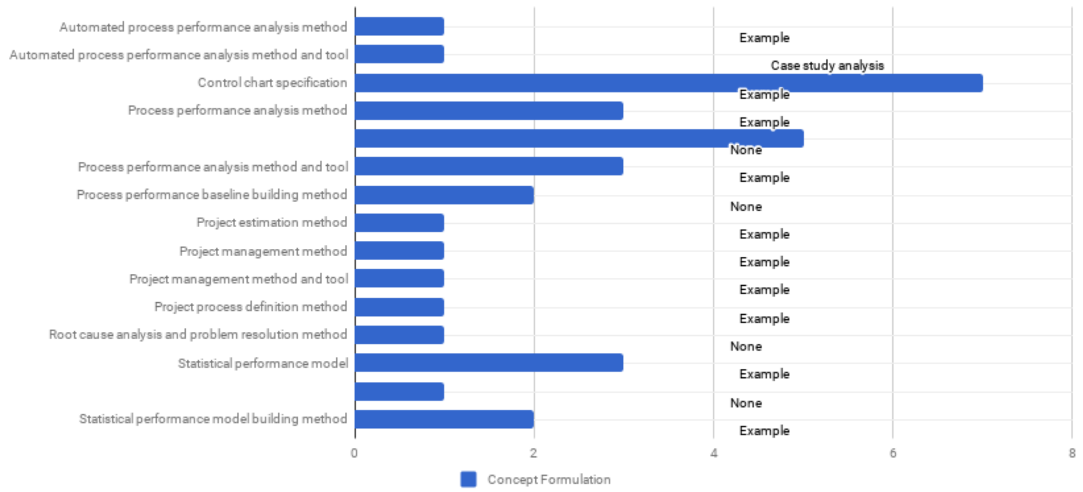

Figure 12. Types of methods, techniques and tools at the technology maturation phase *Concept Formulation* and the method applied to evaluate them

The last three contributions identified are at the software technology maturation phase *Internal Enhancement and Exploration*, meaning they were evaluated through some experiments.

6 Related Work

This section describes related work. We did not find a previous systematic literature mapping (SLM) on the subject, but we did find studies providing knowledge and supporting implementation of high-maturity practices. Dong et al. (2016) propose an organizational process asset library (OPAL) architecture for process improvement, especially for high-maturity process improvement. The authors describe the OPAL architecture components, and its use for high-maturity process improvement in an actual enterprise, stating that the architecture supports easy maintenance and extension. Although this OPAL architecture can support high-maturity project management, such an architecture is expected to be provided by organizations for projects. Therefore, this study was considered to be only at the organizational level, not assisting directly in project management.

Barcellos et al. (2013) present a strategy to help software organizations prepare themselves regarding measurement aspects in order to implement SPC. The strategy is made up of three components. The first component is a reference software measurement ontology, which provides a common vocabulary and relevant knowledge about the software measurement domain, including traditional and high maturity measurement aspects. The second component is an instrument for evaluating the suitability of a measurement repository for SPC, which is used to evaluate existing measurement repositories and to determine their suitability for SPC, identifying corrective actions that can be taken as a means to obtain measurement repositories suitable for SPC (if it is necessary and feasible). The third component is a body of recommendations for software measurement suitable for SPC, which provides guidelines on how to prepare a measurement program, how to define measures, and how to perform measurements suitable for SPC. Since this strategy helps organizations prepare for SPC, this study was considered to be only at the organizational level, not directly assisting in project management.

7 Threats to Validity

This section describes threats to validity which were identified for the present study, along with the strategies applied to mitigate them, as well as aspects which should be considered before generalizing any of its findings. They are organized according to (PETERSEN; VAKKALANKA; KUZNIARZ, 2015).

To mitigate threats concerning *theoretical validity* and potential bias, we involved three researchers in this study. The first author applied both the first and second filter steps. The second author, with more experience in performing systematic literature mappings (SLM), performed the first filter step and discussed with the first author the inclusions and exclusions proposed for the second filter, reading the full text of any papers where there was any disagreement. A third researcher also verified some of the decisions made, but since he was a third person, we judged that it was enough for him to check only the papers which one of us decided to exclude, so that he had to read a smaller amount of papers. If any paper was not selected to be excluded by all three researchers, it was considered as included and further analyzed.

To mitigate threats regarding the search string, we followed a search string elaboration process, applying a control group for the papers as well as PICOC, thus arriving at acceptable precision and sensitivity. In addition, to minimize this threat, the present study closely followed generally accepted practices for SLM (KITCHENHAM;

CHARTERS, 2007; PETERSEN et al., 2008; PETERSEN; VAKKALANKA; KUZNIARZ, 2015).

To mitigate threats concerning *descriptive validity*, the data extraction fields were discussed and defined by both authors prior to the research and refined during it. However, the data extraction was performed only by the first author of this research and may contain errors since some of the papers do not provide clear descriptions or comprehensive information, making data extraction difficult. In addition, after reading some of the papers more than once, we noticed that as we gained more knowledge about the subject, we were better able to analyze the papers for which data had already been extracted. Therefore, some data extraction may have issues related to the lack of statistical knowledge of the first researcher regarding the reported contributions. To mitigate this threat, the second author read the extractions to evaluate their understandability, and to verify their applicability on software engineering area, and their type classification.

Regarding the degree of *generalizability* of the findings, the first identified threat concerns the limitation of this study to four search engines, although experience shows that those search engines do provide good coverage of the area of Software Engineering (KITCHENHAM; CHARTERS, 2007; MATALONGA, Santiago; RODRIGUES; TRAVASSOS, 2017; OLIVEIRA et al., 2017; SILVA; COSTA VALENTIM; CONTE, 2015).

Another threat concerns the limitation of the search period to fifteen years. To mitigate this threat, we included experience reports in our scope. Thus, we identified methods, techniques and tools proposed, evaluated, experimented with and adopted in this period. Moreover, if some method, technique or tool proposed before that period had good acceptance, reaching *External Enhancement and Exploration* technology maturation phase, it would probably be mentioned in the experience reports from the last fifteen years.

It is interesting to note that papers found on the first years of the period (2003 to 2006) are usually smaller, and present weaker evaluations. Figure 13 shows that 18 methods, techniques and tools out of the 24 identified ones, which do not present any evaluation of their application, are on that period. Some of them just present methods, techniques and tools well known on the area, but that by that time were starting to be applied on software development, such as Six Sigma. Furthermore, Redwine and Riddle (1985) says that a technology usually takes between 11 to 23 years from an idea to the point it can be popularized and disseminated to the technical community at large. Therefore, the search period may cover some of those cases.

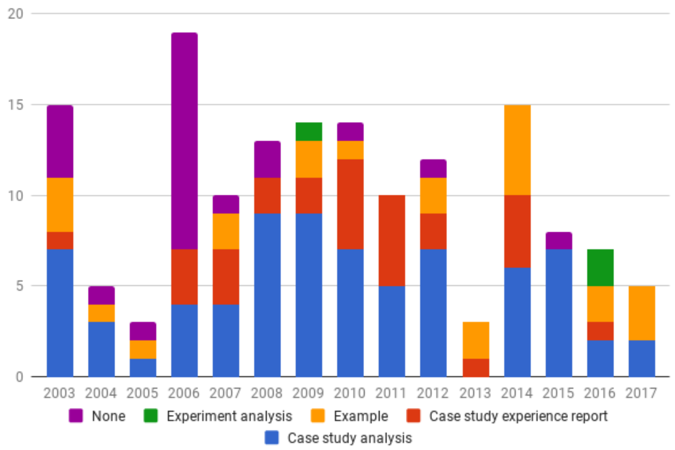

Figure 13. Number of methods, techniques and tools classified by the method applied to evaluate them over the years

8 Conclusions

This paper presents the findings of a Systematic Literature Mapping (SLM) which identified and characterized methods, techniques and tools which can assist in quantitative high-maturity project management. Maturity and capability models such as CMMI mention various methods, techniques and tools (such as control charts or simulations), but they do not provide details about them and do not present the available types of those methods, techniques and tools (such as control chart types or simulation algorithms). Thus, practitioners and researchers usually report a lack of statistical and quantitative knowledge or guidance on how to implement high-maturity practices (CARD; DOMZALSKI; DAVIES, 2008; GONÇALVES, L. et al., 2012; GOU et al., 2009; LEE, Dalju; BAIK; SHIN, 2009; SCHOTS et al., 2014; SHARMA, D. et al., 2016; TAKARA et al., 2007).

We identified 153 methods, techniques and tools which can assist in quantitative high-maturity project management, and classified them into 13 types. Since most of the research questions to characterize those contributions are qualitative, Appendix B provides a table for each type of contribution with short versions of the answers to research questions RQ3 to RQ13. Those tables can be used by practitioners and researchers to get information about the methods, techniques and tools that can be interesting for a particular organizational context or need, like a catalog. Table A1 in Appendix A presents the ID of the paper where the contribution was found, allowing readers to identify papers where the answers are of interesting quality and technology maturation phase. This way, practitioners and researchers can better analyze the contributions identified, and select the ones which best fit their needs. More details about each contribution can be found in the Google spreadsheet and by consulting the papers themselves. The extracted data can also be used as examples to guide the application of the method, technique or tool.

We provide analysis regarding which development methods and characteristics are supported by the contributions identified. In addition, for all the contributions identified, we provide analysis regarding their software technology maturation phase and the method used to evaluate them.

We can summarize some of the findings of our analysis:

(i) Most of the methods, techniques and tools identified are statistical performance models, control chart specifications, and process performance analysis methods, which can be interpreted as an indication that these three types are the ones most used to assist in quantitative high-maturity project management;

(ii) Most of the methods, techniques and tools identified do not specify which development methods and characteristics they support (112 out of 153), which suggest that they could be applicable to any development context;

(iii) There is a trend in researching and adopting methods, techniques and tools which support agile/iterative development, which is in accordance with the trend of agile in software engineering in general and indicates the combination of agile and high-maturity practices;

(iv) There is another trend in researching and adopting methods, techniques and tools which support development for product maintenance/evolution, which indicates the application of high-maturity practices to product evolution, and can be interpreted as high-maturity practices being applied to the area of IT Services;

(v) About half of the methods, techniques and tools identified were evaluated by being applied to historical data in a case study analysis (73 out of 153), which indicates room for improvement in the academic maturity of this research area.

It is our hope that this information can help fill the knowledge gap regarding the actual types of statistical and other quantitative methods, techniques and tools actually being proposed, evaluated, experimented with and adopted by organizations to assist in quantitative high-maturity project management.

Other analysis regarding each type of method, technique and tool, including their descriptions, are going to be provided on papers submitted to journals and conferences, in order to assist on identifying the methods, techniques and tools most appropriate to specific development methods or characteristics, which are in a good technology maturation phase and have been evaluated.

Acknowledgements. The work in this paper was funded by CAPES, FAPERJ (Projects E-26/010.000883/2016, E-26/010.001644/2016), and UNIRIO (Project PQ-UNIRIO 01/2016). The authors also wish to thank Monalessa Barcellos for her support in providing the initial set of papers regarding high-maturity practices which was used to identify the papers in to control group.

Referências Bibliográficas

AGRAWAL, M.; CHARI, K. **Software effort, quality, and cycle time: A study of CMM level 5 projects**. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, [s.l.], v. 33, nº 3, p. 145–156, 2007. ISSN: 0098-5589, DOI: 10.1109/TSE.2007.29.

ALHASSAN, M. A. .; JAWAWI, D. N. A. . **Sequential strategy for software process measurement that uses Statistical Process Control**. JAWAWI D.N.A. SULAIMAN S., M. R. S. N. A. (Org.). In: *2014 8th Malaysian Software Engineering Conference, MySEC 2014*. [s.l.]: Institute of Electrical and Electronics Engineers Inc., 2014. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84929300819&partnerID=40&md5=daca8f3c1311eaae1e79c784d286750f>. ISBN: 9781479954391, DOI: 10.1109/MySec.2014.6985986.

AMAN, H.; OHKOCHI, T. **An application of growth curve model for predicting code churn in open source development**. In: *Proceedings of the 9th Joint Conference on Knowledge-Based Software Engineering, JCKBSE 2010*. Kaunas: [s.n.], 2010. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84856206685&partnerID=40&md5=d0bbfebb00dd97d18f185ccbabf9a369>. ISBN: 9789955258445.

ANIL, R. et al. **A methodology for managing multi-disciplinary programs with six sigma approach**. In: *IEEE International Engineering Management Conference*. Singapore: [s.n.], 2004.

ANTONIOL, G.; GRADARA, S.; VENTURI, G. **Methodological issues in a CMM level 4 implementation**. *Software Process Improvement and Practice*, [s.l.], v. 9, nº 1, p. 33–50, 2004. ISSN: 10774866.

AXELOS. **Managing successful projects with PRINCE2**. 2017 ed. ed. [s.l.]: The Stationery Office, 2017. 400 p. ISBN: 0113315333.

BALDASSARRE, Maria Teresa et al. **Statistically Based Process Monitoring: Lessons from the Trench**. WANG, Q AND GAROUSI, V AND MADACHY, R AND PFAHL, D.

(Org.). In: *Trustworthy Software Development Processes, Proceedings*. [s.l.]: [s.n.], 2009. ISBN: 978-3-642-01679-0, ISSN: 0302-9743.

BALDASSARRE, Maria Teresa; CAIVANO, D.; VISAGGIO, G. **Non invasive monitoring of a distributed maintenance process**. In: *2006 IEEE Instrumentation and Measurement Technology Conference Proceedings, Vols 1-5*. [s.l.]: [s.n.], 2006. ISBN: 978-0-7803-9359-2, ISSN: 1091-5281, DOI: 10.1109/IMTC.2006.328378.

BALDASSARRE, M T et al. **Improving Dynamic Calibration through Statistical Process Control**. In: *ICSM 2005: PROCEEDINGS OF THE 21ST IEEE INTERNATIONAL CONFERENCE ON SOFTWARE MAINTENANCE*. [s.l.]: [s.n.], 2005. ISBN: 0-7695-2368-4, ISSN: 1063-6773.

BARCELLOS, M. P. .; ALMEIDA FALBO, R. . DE; ROCHA, A. R. . **A strategy for preparing software organizations for statistical process control**. *Journal of the Brazilian Computer Society*, [s.l.], v. 19, nº 4, p. 445–473, 2013. ISSN: 01046500, DOI: 10.1007/s13173-013-0106-x.

BARRETO, A. O. S.; ROCHA, A. R. **Defining and monitoring strategically aligned software improvement goals**. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Limerick, v. 6156 LNCS, p. 380–394, 2010. ISBN: 3642137911; 9783642137914, ISSN: 03029743, DOI: 10.1007/978-3-642-13792-1_29.

BASAVARAJ, M. J. .; SHET, K. C. . **Empirical validation of software development effort multipliers of intermediate COCOMO model**. *Journal of Software*, [s.l.], v. 3, nº 5, p. 65–71, 2008a. ISSN: 1796217X.

______. **Estimating and prediction of turn around time for incidents in application service maintenance projects**. *Journal of Software*, [s.l.], v. 3, nº 7, p. 12–21, 2008b. ISSN: 1796217X.

BASILI, V.; CALDIERA, G.; ROMBACH, H. **Goal Question Metric Paradigm**. *Encyclopedia of Software Engineering*, [s.l.], v. 1, p. 528–532, 1994.

BELOW, P. **Forecasting from defect signals**. *CrossTalk*, [s.l.], v. 27, nº 5, p. 38–40, 2014.

BEZERRA, C. I. M. . b et al. **A practical application of performance models to predict the productivity of projects**. In: *Innovations and Advances in Computer Sciences and Engineering*. [s.l.]: [s.n.], 2010. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84883088111&partnerID=40&md5=0e69de5d7828eb229fd8c3f32ac3a293>. ISBN: 9789048136575, DOI: 10.1007/978-90-481-3658-2_47.

BHARATHI, V.; SHASTRY, U. **Neural Network Based Effort Prediction Model for Maintenance Projects**. OCONNOR, RV AND ROUT, T AND MCCAFFERY, F AND DORLING, A. (Org.). In: *Software Process Improvement And Capability Determination*. [s.l.]: [s.n.], 2011. ISBN: 978-3-642-21232-1, ISSN: 1865-0929.

BHARATHI, V.; SHASTRY, U.; RAJ, J. **Bayesian Network Based Bug-fix Effort Prediction Model**. MAS, A AND MESQUIDA, A AND ROUT, T AND OCONNOR, RV AND DORLING, A. (Org.). In: *Software Process Improvement And Capability Determination*. [s.l.]: [s.n.], 2012. ISBN: 978-3-642-30438-5, ISSN: 1865-0929.

BIEHL, R. E. **Six sigma for software**. *IEEE Software*, [s.l.], v. 21, nº 2, p. 68–70, 2004. DOI: 10.1109/MS.2004.1270765.

BIJLSMA, D.; CORREIA, J. P.; VISSER, J. **Automatic Event Detection for Software**

Product Quality Monitoring. FARIA, JP AND SILVA, A AND MACHADO, R. (Org.). In: *2012 Eighth International Conference On The Quality Of Information And Communications Technology (QUATIC 2012)*. [s.l.]: [s.n.], 2012. ISBN: 978-0-7695-4777-0, DOI: 10.1109/QUATIC.2012.22.

BOEHM, B. . et al. **High Maturity is not a procrustean bed**. *CrossTalk*, [s.l.], v. 27, nº 4, p. 8–14, 2014.

BOFFOLI, N. **Non-intrusive monitoring of software quality**. In: *Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR*. [s.l.]: [s.n.], 2006. ISBN: 9780769525365, DOI: 10.1109/CSMR.2006.36.

CANGUSSU, J.W.; DECARLO, R. A.; MATHUR, A. P. **Monitoring the software test process using statistical process control: A logarithmic approach**. In: *Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering*. [s.l.]: [s.n.], 2003. ISBN: 9781581137439, DOI: 10.1145/940071.940093.

CANGUSSU, Joao W. **Integrating statistical and feedback process control for the monitoring of the software test process**. In: *Proceedings of the IASTED International Conference on Software Engineering and Applications*. Marina del Rey, CA, United states: [s.n.], 2003.

CARD, D. N. .; DOMZALSKI, K. .; DAVIES, G. . **Making statistics part of decision making in an engineering organization**. *IEEE Software*, [s.l.], v. 25, nº 3, p. 37–47, 2008. ISSN: 07407459, DOI: 10.1109/MS.2008.66.

CHANG, C.-P.; CHU, C.-P. **Improvement of causal analysis using multivariate statistical process control**. *SOFTWARE QUALITY JOURNAL*, [s.l.], v. 16, nº 3, p. 377–409, 2008. ISSN: 0963-9314, DOI: 10.1007/s11219-007-9042-3.

CHANG, C.-W.; TONG, L.-I. **Monitoring the software development process using a short-run control chart**. *SOFTWARE QUALITY JOURNAL*, [s.l.], v. 21, nº 3, p. 479–499, 2013. ISSN: 0963-9314, DOI: 10.1007/s11219-012-9182-y.

CHEN, T.; ZHOU, B.; LUO, W. **A Process Optimization Method for High Maturity Process Improvements**. In: *Management and Service Science (MASS), 2010 International Conference on*. [s.l.]: [s.n.], 2010. DOI: 10.1109/ICMSS.2010.5577138.

CMMI PRODUCT TEAM. **CMMI® for Development (CMMI-DEV) - Improving processs for developing better products and services, V 1.3, CMU/SEI-2010-TR-033**. [s.l.]: Software Engineering Institute, 2010.

CRAWFORD, J. K. (James K. **Project management maturity model**. 2nd ed. [s.l.]: Auerbach Publications, 2006. 235 p. ISBN: 0849379458.

CUNHA, J. C. . et al. **Implementing software effort estimation in a medium-sized company**. In: *Proceedings - 2011 34th IEEE Software Engineering Workshop, SEW 2011*. Limerick: [s.n.], 2012. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84860006911&partnerID=40&md5=7ff99aad0349fed24f6b3672d6f4bc44>. ISBN: 9780769546278, DOI: 10.1109/SEW.2011.19.

DONG, S.; REN, A.; WANG, X. **The Architecture of OPAL for the Software Process Improvement in High Maturity Level**. In: *2016 Third International Conference on Trustworthy Systems and their Applications (TSA)*. [s.l.]: [s.n.], 2016. DOI: 10.1109/TSA.2016.21.

DONG, S.; REN, A.; WANG, X. **Application of Organizational Process Asset Library**

in High Maturity Process Improvement. In: *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*. [s.l.]: [s.n.], 2017. ISBN: 9781509055272, DOI: 10.1109/ICECCS.2016.039.

EICKELMANN, N.; ANANT, A. **Statistical process control: What you don't measure can hurt you!** *IEEE Software*, [s.l.], v. 20, nº 2, p. 49–51, 2003. DOI: 10.1109/MS.2003.1184166.

FEHLMANN, T. .; KRANICH, E. . **Exponentially weighted moving average (EWMA) prediction in the software development process**. DANEVA M., V. F. (Org.). In: *Proceedings - 2014 Joint Conference of the International Workshop on Software Measurement, IWSM 2014 and the International Conference on Software Process and Product Measurement, Mensura 2014*. [s.l.]: Institute of Electrical and Electronics Engineers Inc., 2014. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84929620370&partnerID=40&md5=02865a92fc92ea4086b5275aeb80f200>. ISBN: 9781479941742, DOI: 10.1109/IWSM.Mensura.2014.50.

FEHLMANN, T. M.; KRANICH, E. **A new approach for continuously monitoring project deadlines in software development**. In: *ACM International Conference Proceeding Series*. [s.l.]: [s.n.], 2017. ISBN: 9781450348539, DOI: 10.1145/3143434.3143439.

FENTON, N. et al. **Making resource decisions for software projects**. In: *Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on*. Edinburgh, United Kingdom: Institute of Electrical and Electronics Engineers Computer Society, Piscataway, NJ 08855-1331, United States, 2004. ISBN: 0270-5257.

FERNANDEZ-CORRALES, C. .; JENKINS, M. .; VILLEGAS, J. . **Application of statistical process control to software defect metrics: An industry experience report**. In: *International Symposium on Empirical Software Engineering and Measurement*. Baltimore, MD: [s.n.], 2013. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84893297356&partnerID=40&md5=69649f70225e700bbda1b785947c993b>. ISSN: 19493770, DOI: 10.1109/ESEM.2013.51.

FERREIRA, A. L. et al. **An Apporach to Improving Software Inspections Performance**. In: *2010 IEEE International Conference On Software Maintenance*. [s.l.]: [s.n.], 2010. ISBN: 978-1-4244-8629-8, ISSN: 1063-6773.

FLORAC, A.; CARLETON, A. D. **Measuring the Software Process: Statistical Process Control for Software Process Improvement**. [s.l.]: Addison-Wesley, 1999.

FRENZ, P J. **Applying measurement principles and adapting a defect predictability model to hardware development**. In: *17th Annual International Symposium of the International Council on Systems Engineering, INCOSE 2007 - Systems Engineering: Key to Intelligent Enterprises*. San Diego, CA: [s.n.], 2007. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84878095629&partnerID=40&md5=99ae2a8f3bedc0b7d045cc29def38de6>. ISBN: 9781605601199.

FRENZ, Paul J; GURVIN, A. C. **Quantitative Analysis: Clawing your way to the top of the maturity pinnacle**. In: *16th Annual International Symposium of the International Council on Systems Engineering, INCOSE 2006*. Orlando, FL, United states: [s.n.], 2006.

GONÇALVES, F. M. G. S. . et al. **Implementing causal analysis and resolution in software development projects: The MiniDMAIC approach**. In: *Proceedings of the Australian Software Engineering Conference, ASWEC*. Perth, WA: [s.n.], 2008. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-

50249094532&partnerID=40&md5=d5880f74dcd3582d435e67ed6226275b>. ISBN: 0769531008; 9780769531007, DOI: 10.1109/ASWEC.2008.4483199.

GONÇALVES, L. et al. **Support for Statistic Process Control of software process**. In: *2012 XXXVIII Conferencia Latinoamericana En Informatica (CLEI)*. [s.l.]: [s.n.], 2012. DOI: 10.1109/CLEI.2012.6426915.

GORDEA, S.; ZANKER, M. **Building maintenance charts and early warning about scheduling problems in software projects**. In: *ICSOFT 2006: Proceedings of the First International Conference on Software and Data Technologies, Vol 1*. [s.l.]: [s.n.], 2006. ISBN: 972-8865-69-4.

GOU, L. . b et al. **Quantitative defects management in iterative development with BiDefect**. *Software Process Improvement and Practice*, [s.l.], v. 14, nº 4, p. 227–241, 2009. ISSN: 10774866, DOI: 10.1002/spip.413.

GROSSI, L.; CALVO-MANZANO, J. A.; SAN FELIU, T. **High-maturity levels: achieving CMMI ML-5 in a consultancy company**. *Journal Of Software-Evolution And Process*, [s.l.], v. 26, nº 9, SI, p. 808–817, 2014. ISSN: 2047-7473, DOI: 10.1002/smr.1666.

HALE, C.; ROWE, M. **Do not get out of control: Achieving real-time quality and performance**. *CrossTalk*, [s.l.], v. 25, nº 1, p. 4–8, 2012.

HALE, J. E.; HALE, D. P. **Evaluating testing effectiveness during software evolution: a time-series cross-section approach**. *Journal Of Software-Evolution And Process*, [s.l.], v. 24, nº 1, p. 35–49, 2012. ISSN: 2047-7481, DOI: 10.1002/smr.531.

HATAMI HARDOROUDI, A. et al. **Robust corrective and preventive action (CAPA)**. In: *2011 IEEE International Systems Conference, SysCon 2011 - Proceedings*. Montreal, QC, Canada: [s.n.], 2011. Disponível em: <http://dx.doi.org/10.1109/SYSCON.2011.5929081>.

HONG, G. Y.; GOH, T. N. **Six Sigma in software quality**. *TQM Magazine*, [s.l.], v. 15, nº 6, p. 364–373, 2003. ISSN: 0954478X.

ISO. **ISO 21500: Guidance on Project Management**. [s.l.]: [s.n.], 2012. Disponível em: <http://www.iso.org/iso/catalogue_detail?csnumber=50003>. ISBN: 9789087538095, ISSN: 02637863, DOI: 10.1016/j.ijproman.2014.10.009.

[CSL STYLE ERROR: reference with no printed form.]

JACOB, A. L.; PILLAI, S. K. **Statistical process control to improve coding and code review**. *IEEE SOFTWARE*, [s.l.], v. 20, nº 3, p. 50+, 2003. ISSN: 0740-7459, DOI: 10.1109/MS.2003.1196321.

JAKOBSEN, C. R. .; POPPENDIECK, T. . **Lean as a Scrum troubleshooter**. In: *Proceedings - 2011 Agile Conference, Agile 2011*. Salt Lake City, UT: [s.n.], 2011. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-80053000049&partnerID=40&md5=8609e13042487b27fe9ba35d26210e83>. ISBN: 9780769543703, DOI: 10.1109/AGILE.2011.11.

JAKOBSEN, C. R.; SUTHERLAND, J. **Scrum and CMMI Going from Good to Great**. In: *2009 Agile Conference*. [s.l.]: [s.n.], 2009. DOI: 10.1109/AGILE.2009.31.

JALOTE, P.; MITTAL, A. K.; PRAJAPAT, R. G. **On optimum module size for software inspections**. In: *International Journal of Reliability, Quality and Safety Engineering*. [s.l.]: [s.n.], 2007. ISSN: 02185393.

KAMMA, D. .; JALOTE, P. . **High productivity programmers use effective task**

processes in unit-testing. PASALA A. SUN J., R. Y. R. B. A. (Org.). In: *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*. [s.l.]: IEEE Computer Society, 2016. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84974687958&partnerID=40&md5=2162cb1611b3d52a16d00158d07a58e7>. ISBN: 9781467396448, ISSN: 15301362, DOI: 10.1109/APSEC.2015.31.

KIM, H.-C. **Assessing software reliability based on NHPP using SPC**. *International Journal of Software Engineering and its Applications*, [s.l.], v. 7, nº 6, p. 61–70, 2013. ISSN: 17389984, DOI: 10.14257/ijseia.2013.7.6.06.

KIMURA, M. .; FUJIWARA, T. . **A new criterion for the optimal software release problems: Moving average quality control chart with bootstrap sampling**. *Communications in Computer and Information Science*, Jeju Island, v. 59 CCIS, p. 280–287, 2009. ISBN: 3642106188; 9783642106187, ISSN: 18650929, DOI: 10.1007/978-3-642-10619-4_34.

KIRBAŞ, S. .; TARHAN, A. .; DEMIRÖRS, O. . **An assessment and analysis tool for statistical process control of software processes**. In: *7th International SPICE Conference on Process Assessment and Improvement, SPICE 2007*. [s.l.]: University of Eastern Finland, 2007. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84946733295&partnerID=40&md5=91902632062d37bb6ffc439f29a7765f>. ISBN: 9788976416094.

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing Systematic Literature Reviews in Software Engineering**. *Engineering*, [s.l.], v. 2, p. 1051, 2007. ISBN: 1595933751, ISSN: 00010782, DOI: 10.1145/1134285.1134500.

KITCHENHAM, B.; JEFFERY, D. R.; CONNAUGHTON, C. **Misleading metrics and unsound analyses**. *IEEE SOFTWARE*, [s.l.], v. 24, nº 2, p. 73+, 2007. ISSN: 0740-7459, DOI: 10.1109/MS.2007.49.

KOJIMA, T. et al. **Risk analysis of software process measurements**. *SOFTWARE QUALITY JOURNAL*, [s.l.], v. 16, nº 3, p. 361–376, 2008. ISSN: 0963-9314, DOI: 10.1007/s11219-007-9040-5.

KOMURO, M. **Experiences of applying SPC techniques to software development processes**. In: *Proceedings - International Conference on Software Engineering*. [s.l.]: [s.n.], 2006. ISBN: 9781595933751.

KOMURO, M.; KOMODA, N. **An Explanation Model for Quality Improvement Effect of Peer Reviews**. In: *Computational Intelligence for Modelling Control Automation, 2008 International Conference on*. [s.l.]: [s.n.], 2008. DOI: 10.1109/CIMCA.2008.187.

KUHRMANN, M. et al. **Software process improvement: where is the evidence? Initial findings from a systematic mapping study**. In: *Proceedings of the 2015 International Conference on Software and System Process - ICSSP 2015*. New York, New York, USA: ACM Press, 2015. Disponível em: <http://dl.acm.org/citation.cfm?id=2785592.2785600>. Acesso em: 20/abr./16. ISBN: 9781450333467, DOI: 10.1145/2785592.2785600.

KUMARI, K. S.; AMULYA, B.; PRASAD, R. S. **Comparative study of Pareto Type II with HLD in assessing the software reliability with order statistics approach using SPC**. In: *Circuit, Power and Computing Technologies (ICCPCT), 2014 International Conference on*. [s.l.]: [s.n.], 2014. DOI: 10.1109/ICCPCT.2014.7054824.

LEE, Dalju; BAIK, J.; SHIN, J.-H. **Software Reliability Assurance Using a Framework in Weapon System Development: A Case Study**. MIAO, H AND HU, G. (Org.). In: *Proceedings Of The 8th IEEE/ACIS International Conference On Computer And Information*

Science. [s.l.]: [s.n.], 2009. ISBN: 978-0-7695-3641-5, DOI: 10.1109/ICIS.2009.168.

LEE, Donghun; CHA, S. K.; LEE, A. H. **A Performance Anomaly Detection and Analysis Framework for DBMS Development**. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, [s.l.], v. 24, nº 8, p. 1345–1360, 2012. ISSN: 1041-4347, DOI: 10.1109/TKDE.2011.88.

LI, Z. et al. **A definition of software process quality based on statistical process control**. In: *Proceedings of the 11th Joint International Computer Conference, JICC 2005*. [s.l.]: [s.n.], 2005. ISBN: 9789812565327.

LUCIA, A. DE; POMPELLA, E.; STEFANUCCI, S. **Assessing the maintenance processes of a software organization: An empirical analysis of a large industrial project**. *Journal of Systems and Software*, [s.l.], v. 65, nº 2, p. 87–103, 2003. DOI: 10.1016/S0164-1212(02)00051-1.

MARANDI, A. K.; KHAN, D. A. **An Impact of Linear Regression Models for Improving the Software Quality with Estimated Cost**. BUYYA R. RAJA K.B., D. S. P. I. S. S. V. K. R. P. L. M. (Org.). In: *Procedia Computer Science*. [s.l.]: Elsevier, 2015. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84944064163&partnerID=40&md5=8fe9137ecde02837256ef49ddb487612>. ISSN: 18770509, DOI: 10.1016/j.procs.2015.06.039.

MATALONGA, Santiago; RODRIGUES, F.; TRAVASSOS, G. H. **Characterizing testing methods for context-aware software systems: Results from a quasi -systematic literature review**. *Journal of Systems and Software*, [s.l.], v. 131, p. 1–21, 2017. ISSN: 01641212, DOI: 10.1016/j.jss.2017.05.048.

MATALONGA, Santiago; SOLARI, M.; SAN FELIU, T. **An empirically validated simulation for understanding the relationship between process conformance and technology skills**. *SOFTWARE QUALITY JOURNAL*, [s.l.], v. 22, nº 4, p. 593–609, 2014. ISSN: 0963-9314, DOI: 10.1007/s11219-013-9214-2.

MATALONGA, S; SAN FELIU, T. **Calculating return on investment of training using process variation**. *IET SOFTWARE*, [s.l.], v. 6, nº 2, p. 140–147, 2012. ISSN: 1751-8806, DOI: 10.1049/iet-sen.2011.0024.

MC NELLIS, T.; HARRINGTON, H. J. **Remember, the (Internet) applet doesn't fall far from the tree**. *TQM Magazine*, [s.l.], v. 15, nº 5, p. 302–315, 2003. ISSN: 0954478X.

MILLER, S. D. et al. **A control-theoretic approach to the management of the software system test phase**. *JOURNAL OF SYSTEMS AND SOFTWARE*, [s.l.], v. 79, nº 11, p. 1486–1503, 2006. ISSN: 0164-1212, DOI: 10.1016/j.jss.2006.03.033.

MOHAN, K. K. et al. **Early Quantitative Software Reliability Prediction Using Petri-nets**. In: *IEEE Region 10 Colloquium And Third International Conference On Industrial And Information Systems, Vols 1 And 2*. 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, 2008. ISBN: 978-1-4244-2805-2.

MOHAN, K. K.; SRIVIDYA, A.; GEDELA, R. K. **Quality of service prediction using fuzzy logic and RUP implementation for process oriented development**. In: *International Journal of Reliability, Quality and Safety Engineering*. [s.l.]: [s.n.], 2008. Disponível em: <http://dx.doi.org/10.1142/S021853930800299X>. ISSN: 02185393.

MOHAPATRA, S. **Improvised process for quality through quantitative project management: An experience from software development projects**. *International Journal of Information and Communication Technology*, [s.l.], v. 2, nº 4, p. 355–373, 2010. ISSN:

14666642, DOI: 10.1504/IJICT.2010.034977.

MONTEIRO, L. F. S.; OLIVEIRA, K. M. DE. **Defining a catalog of indicators to support process performance analysis**. *Journal of Software Maintenance and Evolution*, [s.l.], v. 23, nº 6, p. 395–422, 2011. ISSN: 1532060X, DOI: 10.1002/smr.482.

MURUGAPPAN, M.; KEENI, G. **Blending CMM and Six Sigma to meet business goals**. *IEEE Software*, [s.l.], v. 20, nº 2, p. 42–48, 2003. DOI: 10.1109/MS.2003.1184165.

NAKAMURA, N. . et al. **Approach to introducing a statistical quality control**. In: *Proceedings - Joint Conference of the 21st International Workshop on Software Measurement, IWSM 2011 and the 6th International Conference on Software Process and Product Measurement, MENSURA 2011*. Nara: [s.n.], 2011. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84856185792&partnerID=40&md5=e3037a16cf8bc25c522b3d22fbaed31f>. ISBN: 9780769544977, DOI: 10.1109/IWSM-MENSURA.2011.25.

NANDITHA, J. et al. **Optimized defect prediction model using statistical process control and Correlation-Based feature selection method**. *Advances in Intelligent Systems and Computing*, [s.l.], v. 384, p. 355–366, 2016. ISBN: 9783319230351, ISSN: 21945357, DOI: 10.1007/978-3-319-23036-8_31.

NARAYANA, V.; SWAMY, R. **Experiences in the inspection process characterization techniques**. TITSWORTH, F (Org.). In: *THIRD INTERNATIONAL CONFERENCE ON QUALITY SOFTWARE, PROCEEDINGS*. [s.l.]: [s.n.], 2003. ISBN: 0-7695-2015-4, DOI: 10.1109/QSIC.2003.1319126.

NELSON, L. S. **Technical Aids**. *Journal of Quality Technology*, [s.l.], v. 16, nº 4, p. 238–239, 1984.

NGUYEN, T. H. D. . et al. **Automated detection of performance regressions using statistical process control techniques**. In: *ICPE'12 - Proceedings of the 3rd Joint WOSP/SIPEW International Conference on Performance Engineering*. Boston, MA: [s.n.], 2012. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84861086105&partnerID=40&md5=ecdd7b01fb886d07b3f71d157215acdf>. ISBN: 9781450312028, DOI: 10.1145/2188286.2188344.

OLIVEIRA, E. et al. **How have Software Engineering Researchers been Measuring Software Productivity? - A Systematic Mapping Study**. In: *Proceedings of the 19th International Conference on Enterprise Information Systems*. [s.l.]: SCITEPRESS - Science and Technology Publications, 2017. Disponível em: <http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006314400760087>. ISBN: 978-989-758-247-9, DOI: 10.5220/0006314400760087.

PAI, D. R.; SUBRAMANIAN, G. H.; PENDHARKAR, P. C. **Benchmarking software development productivity of CMMI level 5 projects**. *INFORMATION TECHNOLOGY & MANAGEMENT*, [s.l.], v. 16, nº 3, p. 235–251, 2015. ISSN: 1385-951X, DOI: 10.1007/s10799-015-0234-4.

PANG, K.-P. .; ALI, S. . **Retrospective analysis for mining the causes in manufacturing processes**. In: *CIMCA 2006: International Conference on Computational Intelligence for Modelling, Control and Automation, Jointly with IAWTIC 2006: International Conference on Intelligent Agents Web Technologies ...* Sydney, NSW: [s.n.], 2007. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-38849135683&partnerID=40&md5=113ad9037877df448cb0045ed1955a21>. ISBN: 0769527310; 9780769527314, DOI: 10.1109/CIMCA.2006.186.

PETERSEN, K. et al. **Systematic mapping studies in software engineering**. *EASE'08 Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering*, [s.l.], p. 68–77, 2008. ISBN: 0-7695-2555-5, ISSN: 02181940, DOI: 10.1142/S0218194007003112.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. **Guidelines for conducting systematic mapping studies in software engineering: An update**. In: *Information and Software Technology*. [s.l.]: [s.n.], 2015. ISBN: 0360-1315, ISSN: 09505849, DOI: 10.1016/j.infsof.2015.03.007.

PETTICREW, M.; ROBERTS, H. **Systematic reviews in the social sciences : a practical guide**. [s.l.]: Blackwell Pub, 2006. 336 p. ISBN: 1405121114.

PROJECT MANAGEMENT INSTITUTE. **A guide to the project management body of knowledge (PMBOK guide)**. [s.l.]: [s.n.], 2017. 756 p. ISBN: 1628251840.

RAFFO, D. M.; SETAMANIT, S.-O. **Supporting software process decisions using bi-directional simulation**. *International Journal of Software Engineering and Knowledge Engineering*, [s.l.], v. 13, nº 5, p. 513–530, 2003. DOI: 10.1142/S0218194003001445.

RAMASUBBU, N. et al. **Work dispersion, process-based learning, and offshore software development performance**. *MIS QUARTERLY*, [s.l.], v. 32, nº 2, p. 437–458, 2008. ISSN: 0276-7783.

RAMASUBBU, N.; BALAN, R. K. **The Impact of Process Choice in High Maturity Environments: An Empirical Analysis**. In: *2009 31st International Conference On Software Engineering, Proceedings*. [s.l.]: [s.n.], 2009. ISBN: 978-1-4244-3452-7, ISSN: 0270-5257, DOI: 10.1109/ICSE.2009.5070551.

RAVI, S. P.; SUPRIYA, N.; KRISHNA MOHAN, G. **SPC for Software Reliability: Imperfect software debugging model**. *International Journal of Computer Science Issues*, [s.l.], v. 8, nº 3 3-2, p. 219–224, 2011. ISSN: 16940814.

RAZA, M.; FARIA, J. P. **A model for analyzing performance problems and root causes in the personal software process**. *JOURNAL OF SOFTWARE-EVOLUTION AND PROCESS*, 111 RIVER ST, HOBOKEN 07030-5774, NJ USA, v. 28, nº 4, SI, p. 254–271, 2016a. ISSN: 2047-7473, DOI: 10.1002/smr.1759.

______. **ProcessPAIR: A tool for automated performance analysis and improvement recommendation in software development**. In: *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)*. [s.l.]: [s.n.], 2016b.

REDWINE, S. T.; RIDDLE, W. E. **Software Technology Maturation**. *Proceedings of the 8th international conference on Software engineering ICSE '85*, [s.l.], p. 189–200, 1985. ISBN: 0818606207, ISSN: 02705257.

ROESELER, A.; PECAK, M.; SHIFFMAN, N. **Using Statistical Process Control to improve the quality and delivery of IT services**. In: *36th International Conference Computer Measurement Group*. Orlando, FL: [s.n.], 2010. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84872145978&partnerID=40&md5=19c1c2d0a481bf46c38ce4a276cb0e2c>.

SÁNCHEZ-ROSADO, I. . b et al. **Assessing the Documentation Development Effort in Software Projects**. ABRAN, A AND BRAUNGARTEN, R AND DUMKE, RR AND CUADRADO GALLEGO, JJ AND BRUNEKREEF, J. (Org.). In: *Software Process And Product Measurement, Proceedings*. [s.l.]: [s.n.], 2009. ISBN: 978-3-642-05414-3, ISSN: 0302-9743.

SARANG, N.; SANGLIKAR, M. A. **An Analysis of Effort Variance in Software Maintenance Projects**. SOBH, T (Org.). In: *Advances in Computer and Informatiom Sciences and Engineering*. [s.l.]: [s.n.], 2008. ISBN: 978-1-4020-8740-0, DOI: 10.1007/978-1-4020-8741-7_66.

SARGUT, K. U.; DEMIRÖRS, O. **Utilization of statistical process control (SPC) in emergent software organizations: Pitfalls and suggestions**. *Software Quality Journal*, [s.l.], v. 14, nº 2, p. 135–157, 2006. DOI: 10.1007/s11219-006-7599-x.

SCHNEIDEWIND, N. **What can software engineers learn from manufacturing to improve software process and product?** *Journal of Intelligent Manufacturing*, [s.l.], v. 22, nº 4, p. 597–606, 2011. ISSN: 09565515, DOI: 10.1007/s10845-009-0322-6.

SCHOTS, N. C. L. . et al. **Supporting software process performance analysis through a knowledge-based environment**. EZZATTI P., D. A. (Org.). In: *Proceedings of the 2014 Latin American Computing Conference, CLEI 2014*. [s.l.]: Institute of Electrical and Electronics Engineers Inc., 2014. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84919459101&partnerID=40&md5=46ab225fa060043f66ca39ebc4d215be>. ISBN: 9781479961306, DOI: 10.1109/CLEI.2014.6965146.

SESHAGIRI, G. **High maturity pays off it is hard to believe unless you do it**. *CrossTalk*, [s.l.], v. 25, nº 1, p. 9–14, 2012.

SHARMA, B.; NAG, R.; MAKKAD, M. **Process Performance Models in Software Engineering: A Mathematical Solution Approach to Problem Using Industry Data**. *Wireless Personal Communications*, [s.l.], v. 97, nº 4, p. 5367–5384, 2017. DOI: 10.1007/s11277-017-4783-1.

SHARMA, D. et al. **Agile 5 using high maturity CMMI practices to improve agile processes and achieve predictable results**. *CrossTalk*, [s.l.], v. 29, nº 4, p. 32–35, 2016.

SHAW, M.; SHAW, M. **Writing Good Software Engineering Research Papers**. *Proceedings of 25th International Conference on Software Engineering (ICSE'03)*, [s.l.], p. 726–736, 2003. ISBN: 0-7695-1877-X, ISSN: 02705257, DOI: 10.1109/ICSE.2003.1201262.

SHEWHART, W. **The Economic Control of Quality of Manufactured Product**. New York: D. Van Nostrand Company, reprinted by ASQC Quality Press, Milwaukee, Wisconsin, 1980, 1931.

SILVA, W.; COSTA VALENTIM, N. M.; CONTE, T. **Integrating the Usability into the Software Development Process - A Systematic Mapping Study**. In: *Proceedings of the 17th International Conference on Enterprise Information Systems*. [s.l.]: SCITEPRESS - Science and and Technology Publications, 2015. Disponível em: <http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0005377701050113>. ISBN: 978-989-758-096-3, DOI: 10.5220/0005377701050113.

SOUZA, A. D. . DE; ROCHA, A. R. .; SANTOS, D. C. S. . DOS. **A proposal for the improvement of project's cost predictability using earned value management and historical data of cost - An empirical study**. In: *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*. [s.l.]: Knowledge Systems Institute Graduate School, 2014. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84938380714&partnerID=40&md5=5214b5e3ce88b1f2894786be07ee1149>. ISSN: 23259000.

TAKARA, A. et al. **Problems and Pitfalls in a CMMI level 3 to level 4 Migration**

Process. BETTIN, A. X. (Org.). In: *Quality of Information and Communications Technology, 2007. QUATIC 2007. 6th International Conference on the*. Lisboa, Portugal: [s.n.], 2007.

TARHAN, A.; DEMIRORS, O. **Apply Quantitative Management Now**. *IEEE SOFTWARE*, [s.l.], v. 29, nº 3, p. 77–85, 2012. ISSN: 0740-7459.

VIJAYA, G. .; ARUMUGAM, S. . **Monitoring the stability of the processes in defined level software companies using control charts with three sigma limits**. *WSEAS Transactions on Information Science and Applications*, [s.l.], v. 7, nº 9, p. 1200–1209, 2010. ISSN: 17900832.

WALLSHEIN, C. C.; LOERCH, A. G. **Software cost estimating for CMMI Level 5 developers**. *JOURNAL OF SYSTEMS AND SOFTWARE*, [s.l.], v. 105, p. 72–78, 2015. ISSN: 0164-1212, DOI: 10.1016/j.jss.2015.03.069.

WANG, Q. et al. **BSR: A statistic-based approach for establishing and refining software process performance baseline**. In: *Proceedings - International Conference on Software Engineering*. [s.l.]: [s.n.], 2006. ISBN: 9781595933751.

WANG, Q.; LI, M. S. **Measuring and improving software process in China**. In: *2005 International Symposium on Empirical Software Engineering (ISESE), Proceedings*. [s.l.]: [s.n.], 2005. ISBN: 0-7803-9507-7.

WEBB, D. R.; MILUK, G.; BUREN, J. VAN. **CMMI level 5 and the team software process**. *CrossTalk*, [s.l.], v. 20, nº 4, p. 16–20, 2007.

WENJIE, L. et al. **Research on CMMI-based Project Management Environment**. In: *2008 4th International Conference On Wireless Communications, Networking And Mobile Computing, Vols 1-31*. [s.l.]: [s.n.], 2008. ISBN: 978-1-4244-2107-7.

WHEELER, D. J. **Understanding Variation - The Key to Managing Chaos**. 2nd ed. [s.l.]: SPC PRESS (Statistical Process Control); 2 Revised edition (September 4, 2000), 2000. 174 p. ISBN: 0945320531.

YAHYA, M. AL; AHMAD, R.; LEE, S. **Impact of CMMI Based Software Process Maturity on COCOMO II's Effort Estimation**. *INTERNATIONAL ARAB JOURNAL OF INFORMATION TECHNOLOGY*, [s.l.], v. 7, nº 2, p. 129–137, 2010. ISSN: 1683-3198.

YAMADA, S.; KII, R. **Software quality analysis for agile development**. In: *2015 4th International Conference on Reliability, Infocom Technologies and Optimization: Trends and Future Directions, ICRITO 2015*. [s.l.]: Institute of Electrical and Electronics Engineers Inc., 2015. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84961777949&partnerID=40&md5=518f382bb5e7c65f842406caad06900e>. ISBN: 9781467372312, DOI: 10.1109/ICRITO.2015.7359201.

YAMADA, S.; YAMAGUCHI, M. **A Method of Statistical Process Control for Successful Open Source Software Projects and Its Application to Determining the Development Period**. *International Journal of Reliability, Quality and Safety Engineering*, [s.l.], v. 23, nº 5, 2016. DOI: 10.1142/S0218539316500182.

ZHANG, H.; KIM, S. **Monitoring Software Quality Evolution for Defects**. *IEEE SOFTWARE*, [s.l.], v. 27, nº 4, p. 58–64, 2010. ISSN: 0740-7459, DOI: 10.1109/MS.2010.66.

ZHANG, S. . b; WANG, Y.-J. . c; RUAN, L. . **Personal software process capability assessment method**. *Ruan Jian Xue Bao/Journal of Software*, [s.l.], v. 20, nº 12, p. 3137–3149, 2009. ISSN: 10009825, DOI: 10.3724/SP.J.1001.2009.00582.

ZHANG, Y. F.; SHETH, D. **Mining software repositories for model-driven development**. *IEEE SOFTWARE*, [s.l.], v. 23, nº 1, p. 82+, 2006. ISSN: 0740-7459, DOI:

10.1109/MS.2006.23.

ZHAO, F.; PENG, X.; ZHAO, W. **Software Development Process Monitoring Based on Nominal Transformation**. MIAO, H AND HU, G. (Org.). In: *Proceedings Of The 8th IEEE/ACIS International Conference On Computer And Information Science*. [s.l.]: [s.n.], 2009. ISBN: 978-0-7695-3641-5, DOI: 10.1109/ICIS.2009.81.

ZHU, M. . et al. **Target based software process evaluation model and application**. In: *2009 2nd International Conference on Information and Computing Science, ICIC 2009*. Manchester: [s.n.], 2009. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-70449510206&partnerID=40&md5=b9723c7a58fe1603d5c6fd52a46c6b63>. ISBN: 9780769536347, DOI: 10.1109/ICIC.2009.34.

Apêndice 1 Quality criteria, quality score and software technology maturation phase of selected papers

Table A1. IDs, references, answers to quality criteria (QC), quality score and software technology maturation phase of selected papers

ID	Reference	QC1	QC2	QC3	QC4	QC5	Quality score	Phase of Software Technology Maturation
2017.03	(DONG; REN; WANG, 2017)	1	3	4	4	3	15	Development and Extension
2017.04	(FEHLMANN, T. M.; KRANICH, 2017)	2	3	2	1	3	11	Concept Formulation
2017.11	(SHARMA, B.; NAG; MAKKAD, 2017)	1	3	2	1	3	10	Concept Formulation
2016.03	(KAMMA; JALOTE, 2016)	2	3	5	5	5	20	Internal Enhancement and Exploration
2016.07	(NANDITHA et al., 2016)	1	3	5	4	3	16	Development and Extension
2016.10	(RAZA; FARIA, 2016a)	2	3	4	4	3	17	Development and Extension
2016.11	(RAZA; FARIA, 2016b)	1	3	4	5	4	17	Internal Enhancement and Exploration
2016.13	(SHARMA, D. et al., 2016)	2	1	3	4	1	11	External Enhancement and Exploration
2016.17	(YAMADA; YAMAGUCHI, 2016)	1	3	2	1	3	10	Concept Formulation
2015.09	(MARANDI; KHAN, 2015)	2	3	1	1	3	10	Concept Formulation
2015.10	(PAI; SUBRAMANIAN; PENDHARKAR, 2015)	1	3	4	4	3	16	Development and Extension
2015.13	(WALLSHEIN; LOERCH, 2015)	2	3	5	4	3	17	Development and Extension
2015.14	(YAMADA; KII, 2015)	2	3	4	4	3	16	Development and Extension
2014.01	(ALHASSAN; JAWAWI, 2014)	1	2	4	4	5	16	External Enhancement and Exploration
2014.02	(BELOW, 2014)	2	2	2	1	3	10	Concept Formulation
2014.03	(BOEHM et al., 2014)	1	3	2	1	3	10	Concept Formulation
2014.06	(SOUZA, DE; ROCHA; SANTOS, DOS, 2014)	2	3	4	4	3	16	Development and Extension
2014.08	(FEHLMANN, T. .; KRANICH, 2014)	2	3	5	4	3	17	Development and Extension
2014.12	(GROSSI; CALVO-MANZANO; SAN FELIU, 2014)	2	2	3	4	5	16	External Enhancement and Exploration
2014.21	(KUMARI; AMULYA; PRASAD, 2014)	2	3	4	4	3	16	Development and Extension
2014.24	(MATALONGA, Santiago; SOLARI; SAN FELIU, 2014)	2	3	5	4	3	17	Development and Extension
2014.29	(SCHOTS et al., 2014)	2	3	2	1	3	11	Concept Formulation
2013.04	(CHANG, C.-W.; TONG, 2013)	2	3	2	1	3	11	Concept Formulation
2013.08	(FERNANDEZ-CORRALES; JENKINS; VILLEGAS, 2013) ★	2	2	3	4	5	16	External Enhancement and Exploration
2013.12	(KIM, 2013) ♦	1	2	2	1	3	9	Concept Formulation
2012.05	(BHARATHI; SHASTRY; RAJ, 2012)	1	3	5	4	3	16	Development and Extension
2012.06	(BIJLSMA; CORREIA; VISSER, 2012)	1	3	5	4	5	18	External Enhancement and Exploration

ID	Reference	QC1	QC2	QC3	QC4	QC5	Quality score	Phase of Software Technology Maturation
2012.10	(CUNHA et al., 2012)	1	2	3	4	5	15	External Enhancement and Exploration
2012.14	(GONÇALVES, L. et al., 2012)	1	3	2	1	3	10	Concept Formulation
2012.15	(HALE, C.; ROWE, 2012)	1	2	2	1	5	11	External Enhancement and Exploration
2012.16	(HALE, J. E.; HALE, 2012)	2	3	5	4	5	19	External Enhancement and Exploration
2012.20	(LEE, Donghun; CHA; LEE, 2012)	1	3	5	4	5	18	External Enhancement and Exploration
2012.22	(MATALONGA, S; SAN FELIU, 2012)	2	3	4	4	5	18	External Enhancement and Exploration
2012.24	(NGUYEN et al., 2012)	1	3	5	4	3	16	Development and Extension
2012.27	(SESHAGIRI, 2012)	1	2	1	1	5	10	External Enhancement and Exploration
2012.30	(TARHAN; DEMIRORS, 2012) ★	1	3	4	4	3	15	Development and Extension
2011.01	(BHARATHI; SHASTRY, 2011)	1	3	5	4	3	16	Development and Extension
2011.06	(HATAMI HARDOROUDI et al., 2011)	2	3	4	4	3	16	Development and Extension
2011.10	(JAKOBSEN, C. R. .; POPPENDIECK, 2011)	1	2	2	4	6	15	External Enhancement and Exploration
2011.14	(MONTEIRO; OLIVEIRA, DE, 2011) ★	2	3	4	4	5	18	External Enhancement and Exploration
2011.17	(NAKAMURA et al., 2011) ★	2	2	4	4	6	18	External Enhancement and Exploration
2011.20	(RAVI; SUPRIYA; KRISHNA MOHAN, 2011)	2	3	4	4	3	16	Development and Extension
2011.23	(SCHNEIDEWIND, 2011)	2	3	2	4	3	14	Development and Extension
2010.02	(YAHYA, AL; AHMAD; LEE, 2010)	2	3	5	4	3	17	Development and Extension
2010.03	(AMAN; OHKOCHI, 2010)	1	3	5	4	3	16	Development and Extension
2010.07	(BARRETO; ROCHA, 2010) ★	2	3	4	4	4	17	Concept Formulation
2010.09	(BEZERRA et al., 2010)	1	3	5	4	5	18	External Enhancement and Exploration
2010.12	(CHEN; ZHOU; LUO, 2010) ♦	1	3	1	1	3	9	Concept Formulation
2010.14	(FERREIRA et al., 2010)	2	3	5	4	5	19	External Enhancement and Exploration
2010.19	(MOHAPATRA, 2010)	1	3	2	1	3	10	Concept Formulation
2010.21	(ROESELER; PECAK; SHIFFMAN, 2010)	1	2	3	4	5	15	External Enhancement and Exploration
2010.26	(VIJAYA; ARUMUGAM, 2010) ★	1	2	3	4	3	13	Development and Extension
2010.29	(ZHANG, H.; KIM, 2010)	2	1	2	4	3	12	Development and Extension
2009.02	(BALDASSARRE, Maria Teresa et al., 2009)	2	3	4	4	3	16	Development and Extension
2009.07	(GOU et al., 2009) ★	1	3	4	4	5	17	External Enhancement and Exploration
2009.09	(JAKOBSEN, C. R.; SUTHERLAND, 2009)	1	2	3	4	5	15	External Enhancement and Exploration
2009.11	(KIMURA; FUJIWARA, 2009)	1	3	2	1	3	10	Concept Formulation
2009.12	(LEE, Dalju; BAIK; SHIN, 2009)	2	3	4	4	5	18	External Enhancement and Exploration
2009.17	(RAMASUBBU; BALAN, 2009)	2	3	5	4	3	17	Development and Extension
2009.18	(SÁNCHEZ-ROSADO et al., 2009)	2	3	4	5	4	18	Internal Enhancement and Exploration
2009.26	(ZHANG, S. . b; WANG; RUAN, 2009)	1	3	4	4	3	15	Development and Extension

ID	Reference	QC1	QC2	QC3	QC4	QC5	Quality score	Phase of Software Technology Maturation
2009.27	(ZHAO; PENG; ZHAO, 2009)	1	3	2	1	3	10	Concept Formulation
2009.28	(ZHU et al., 2009)	1	3	4	4	3	15	Development and Extension
2008.03	(BASAVARAJ; SHET, 2008b)	2	2	3	4	1	12	External Enhancement and Exploration
2008.04	(BASAVARAJ; SHET, 2008a)	2	3	4	4	3	16	Development and Extension
2008.07	(CARD; DOMZALSKI; DAVIES, 2008)	1	2	3	4	1	11	External Enhancement and Exploration
2008.09	(CHANG, C.-P.; CHU, 2008)	2	3	4	4	3	16	Development and Extension
2008.14	(GONÇALVES, F. M. G. S. . et al., 2008)	2	3	1	1	3	10	Concept Formulation
2008.20	(KOJIMA et al., 2008)	2	3	5	4	3	17	Development and Extension
2008.21	(KOMURO; KOMODA, 2008)	1	3	5	4	3	16	Development and Extension
2008.25	(MOHAN et al., 2008)	2	3	4	4	3	16	Development and Extension
2008.26	(MOHAN; SRIVIDYA; GEDELA, 2008)	2	3	4	4	3	16	Development and Extension
2008.29	(RAMASUBBU et al., 2008)	2	3	5	4	3	17	Development and Extension
2008.31	(SARANG; SANGLIKAR, 2008)	2	3	5	4	5	19	External Enhancement and Exploration
2008.39	(WENJIE et al., 2008)	1	3	1	1	6	12	External Enhancement and Exploration
2007.01	(AGRAWAL; CHARI, 2007)	1	3	5	4	3	16	Development and Extension
2007.09	(FRENZ, P J, 2007)	1	2	3	4	1	11	External Enhancement and Exploration
2007.13	(JALOTE; MITTAL; PRAJAPAT, 2007)	2	3	2	4	3	14	Concept Formulation
2007.14	(KIRBAŞ; TARHAN; DEMIRÖRS, 2007)	1	2	4	4	3	14	Development and Extension
2007.15	(KITCHENHAM; JEFFERY; CONNAUGHTON, 2007)	1	2	3	4	1	11	Development and Extension
2007.19	(PANG; ALI, 2007)	1	3	2	1	3	10	Concept Formulation
2007.26	(TAKARA et al., 2007) ♦	1	1	1	1	1	5	External Enhancement and Exploration
2007.29	(WEBB; MILUK; BUREN, VAN, 2007)	1	2	3	4	5	15	External Enhancement and Exploration
2006.03	(BALDASSARRE, Maria Teresa; CAIVANO; VISAGGIO, 2006)	2	3	1	1	3	10	Concept Formulation
2006.05	(BOFFOLI, 2006) ♦	1	3	4	4	5	17	External Enhancement and Exploration
2006.09	(FRENZ, Paul J; GURVIN, 2006)	1	2	3	1	1	8	External Enhancement and Exploration
2006.12	(GORDEA; ZANKER, 2006)	1	3	5	4	3	16	Development and Extension
2006.16	(KOMURO, 2006) ♦	1	2	3	1	1	8	External Enhancement and Exploration
2006.22	(MILLER et al., 2006)	2	3	4	4	3	16	Development and Extension
2006.24	(SARGUT; DEMIRÖRS, 2006)	2	2	3	4	1	12	Development and Extension
2006.27	(WANG et al., 2006)	1	3	4	4	6	18	External Enhancement and Exploration
2006.28	(ZHANG, Y. F.; SHETH, 2006) ♦	1	2	3	1	1	8	External Enhancement and Exploration
2005.01	(BALDASSARRE, M T et al., 2005)	2	3	4	4	3	16	External Enhancement and Exploration
2005.11	(LI et al., 2005) ♦	1	3	1	1	3	9	Concept Formulation
2005.14	(WANG; LI, 2005)	1	3	2	1	3	10	Concept Formulation
2004.01	(ANIL et al., 2004)	1	3	2	1	3	10	Concept Formulation

ID	Reference	QC1	QC2	QC3	QC4	QC5	Quality score	Phase of Software Technology Maturation
2004.02	(ANTONIOL; GRADARA; VENTURI, 2004)	2	2	4	4	5	17	External Enhancement and Exploration
2004.04	(BIEHL, 2004) ♦	1	3	1	1	3	9	Concept Formulation
2003.02	(CANGUSSU, Joao W, 2003)	1	3	2	1	3	10	Concept Formulation
2003.03	(CANGUSSU, J.W.; DECARLO; MATHUR, 2003)	1	3	2	1	3	10	Concept Formulation
2003.05	(LUCIA, DE; POMPELLA; STEFANUCCI, 2003)	2	3	5	4	3	17	Development and Extension
2003.07	(EICKELMANN; ANANT, 2003)	1	3	2	1	3	10	Concept Formulation
2003.11	(HONG; GOH, 2003) ♦	1	3	1	1	3	9	Concept Formulation
2003.12	(JACOB; PILLAI, 2003) ♦	1	2	3	1	1	8	External Enhancement and Exploration
2003.16	(MC NELLIS; HARRINGTON, 2003)	2	3	4	4	5	18	External Enhancement and Exploration
2003.17	(MURUGAPPAN; KEENI, 2003)	1	2	3	4	5	15	External Enhancement and Exploration
2003.18	(NARAYANA; SWAMY, 2003) ♦	1	2	3	1	1	8	Concept Formulation
2003.19	(RAFFO; SETAMANIT, 2003)	2	3	4	4	3	16	Development and Extension

Table labels:

★ Control group paper

♦ Low quality score paper

Apêndice 2 Answers to research questions (RQ)

Table B1. Statistical performance model building methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2017. 11	Time series Analysis, Holt-Winter's method, Holt's smoothing method, Solver tool in Excel.	They deal with problems related to time series and build process performance models.	Inputs: Historical data. Outputs: Number of defects arriving per week estimation.	Number of defects arriving per week.	Not specified.	-	Example.	-	The solution worked out for the stated problem. It can lead to the solution to similar problems using either regression technique, Bayesian network or other models.	No.	–
2017. 11	Queuing theory, Poisson queuing system, Holt-winters method, Solver tool in Excel.	They deal with problems related queuing theory and build process performance models.	Inputs: Historical data on queuing data. Outputs: Mathematical relation between the wait time and the service time.	Wait time, service time.	Not specified.	-	Example.	-	The solution worked out for the stated problem. It can lead to the solution to similar problems using either regression technique, Bayesian network or other models.	No.	–
2016. 03	Markov chains, Euclidean distance.	They use Markov chains to model a task process. They use Euclidean distance to measure the difference between task processes.	Inputs: Video recordings of their tasks. Outputs: Markov chains for each programmer, difference between the chains.	Testing productivity, task steps, probabilities between steps.	Not specified.	18 programmers' videos of at least 2 tasks in 3 model based unit-testing projects.	A task process for model-based unit-testing was created and the steps were verified with the programmers.	–	High-productivity programmers' task processes can be used to teach low-productivity ones.	Yes.	High-productivity programmers' task processes are similar while low-productivity ones are different.
2016. 07	Pearson correlation, ANOVA, SPC.	They use Pearson correlation and ANOVA to select the attributes. They use control charts to generate rules for the model.	Inputs: Dataset of project historical data. Outputs: Prediction model with derived rules.	Difficulty, unique operands, unique operators, intelligence, blank lines, design complexity, cyclomatic complexity, branch count.	Not specified.	A software defect dataset with 2,109 records and 22 attributes.	They compared the accuracy of their model with naïve Bayes and J48 for predicting defects on the same dataset.	With existing classification models Naïve Bayes and J48.	Proposed model shows more accuracy than benchmark classification algorithms.	No.	–
2016. 13	Discrete event simulation, causal analysis and resolution.	They use DES to model the sprint process. They use CAR to refine and improve the process.	Inputs: Data from one agile project. Outputs: Performance models to statistically predict the number of story points that will be delivered in the sprint.	Development / test / test case development time, defect density, number of user stories / story points, resource availability.	Agile.	Data from each sprint.	They used it in a project.	–	They achieved better sprint velocity, more requirements per release, lower production defect density, productivity gain and better team satisfaction.	Yes, one large agile project.	They could predict the number of story points likely to be completed during each sprint and release and perform what-if analysis.
2014. 12	Correlation, regression, logic regression, ANOVA, MANOVA, dummy variable regression, chi-square and logit.	They suggest when to use each statistical approach.	Inputs: Historical data. Outputs: Performance models using statistical techniques for predictive / output continuous / discrete variables.	Density of injected defects, percentage of defects removed / removing efficiency / defect correction effort by phase, injected / removed / escaped defects by phase, total effort by phase.	Not specified.	No detail.	None.	–	Project managers use predictive models to consolidate the process defined and to control the probability to achieve the objective.	Yes, but they do not detail it.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2012.05	Bayesian belief networks.	Its use in software predictions.	Inputs: Defect complexity, team experience, analysis effort. Outputs: A network that can predict bug-fix effort after learning the parameters.	Effort spent on bug fix, bug complexity, team experience, analysis effort.	Not specified.	Dataset from telematics area of Automotive domain.	Accuracy and sensitivity.	With real values from the organization.	Accuracy is 75% and sensitivity is 67%. They performed what-if analysis regarding analysis effort and team experience.	No.	–
2011.23	Taguchi methods, Schneidewind software reliability model (SSRM), regression equations.	They use cumulative failure data to compute a loss function and signal-to-noise ratio. They use SSRM to compute predicted cumulative failure.	Inputs: Actual or predicted cumulative software failures, target values of cumulative software failures. Outputs: Loss function, signal-to-noise ratio, two regression equations to predict failure.	Defects over time.	Not specified.	Shuttle failure data from NASA.	They used the method with historical data.	–	Loss functions show there is excessive variation between desired and target values. Signal-to-noise ratios are high.	No.	–
2009.17	Regression analysis.	They model the variables that influence the decision between agile or traditional methodologies.	Inputs: Historical data. Outputs: A model that estimates the probability of a project team adopting a non-standard process.	Client-specific knowledge, extent of client involvement, design and technology newness, estimated project effort, allocated team size, estimated code size.	Not specified.	Data from 112 software project processes and performance from two different CMM and People CMM level 5 companies.	The model's chi-square statistic value is significant, indicating that the model is statistically valid.	They use the model to find similar projects in group and compare their performance indicators.	Projects that adopted a non-standard development processes performed better on productivity, reuse and effort to fix defects. However, they showed an increase in defect density.	No.	–
2008.20	RATS risk system, multiple regression analysis, central limit theorem, non-parametric permutation.	They propose transforming the continuous endpoint, such as profit, to a binary variable such as "Yore".	Inputs: Data from the first month related to quality, productivity, risk. Outputs: A model that predicts risk failure of a project based on the profit rate (Yore = 1 when a project fails and Yore = 0 otherwise).	Productivity, profit rate.	Not specified.	48 data projects from RATS database.	Scatterplot of the cumulative estimated probability vs. the cumulative observed frequency of Yore = 1. Values agreed well, indicating the appropriateness of the model.	With real values.	To apply this, first estimate the risk of Yore for each project to identify those at higher risk. Then, perform any action to reduce risk and re-evaluate using data collected after the treatment.	No.	–
2008.26	RUP, Fuzzy logic.	They used them together.	Inputs: Historical data on requirements, design, coding, unit testing, IST testing. Outputs: The expected number of defects before the beginning of the project.	Percentage of the effort spent on Requirements/Design/ Coding/Unit testing/ IST Testing, the expected number of defects.	RUP.	An analysis was performed on three different modules over three cycles/builds.	They compare the initial estimates against the real number of defects for one module.	With values obtained during testing procedure.	A close match between the experimental results from and the fuzzy prediction approach.	No.	–
2005.01	A previous proposed Dynamic Calibration (DC) approach.	They have integrated SPC with DC, as decision support tool for identifying process performance changes, and for suggesting when to recalibrate the estimation model.	Inputs: A baseline estimation model of the expected effort. Outputs: Model with recalibration as needed.	Developer's performance in LOC/hour.	Renewal project development.	-	Simulation of the approach on a legacy data set of a renewal project, where the induced process improvements made were known.	The error in reverse engineering and restoration project, DC and DC-SPC during simulation.	DC-SPC was able to point out not only all the known process performance changes, but also further changes that we don't know of.	No.	–

Table B2. Statistical performance models: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2016. 10	Literature review, PSP specifications, previous version of the model.	They removed several factors with weak correlation with the top-level Performance indicators (PIs) and added new attributes to support a ranking approach.	Inputs: PSP historical data. Outputs: PIs, their relationships, recommended ranges, approximate statistical distributions and sensitivity coefficient.	The three top-level PIs refer to predictability, quality and productivity.	PSP.	PSP dataset from SEI with 31,140 projects concluded by 3,114 engineers during 295 classes of PSP for Engineers I / II training courses.	Pearson and Spearman correlation coefficients. A case study with 7 projects performed by a PSP developer during the PSP Fundamentals and Advanced training.	–	Model-based automatic analysis can point out problematic areas to focus on in subsequent manual analysis.	No.	–
2016. 17	NHPP Logarithmic Poisson execution time model.	They applied it to defects found in issue tracking systems for open-source product development.	Inputs: Data on defects in the issue tracking system over time. Outputs: Additional development time for attaining the failure intensity target.	Number of defects in issue tracking systems over time.	Open-source software (OSS).	Data from Android and Thunderbird open-source issue tracking systems.	One example.	–	They can estimate the additional development time for attaining the software failure intensity objective.	No.	–
2015. 09	Least-squares minimum regression.	None, merely its use.	Inputs: Number of defects injected in previous phases, number of defects detected during the phase, count of defects removed during the phase. Outputs: Number of defects in the released software.	Number of defects injected, detected, removed and remaining in each phase.	Not specified.	Several projects from various service-based and product-based organizations.	None.	–	Averages below 95% in cumulative statistical analysis of defect removal effectiveness are not adequate in software quality.	No.	–
2015. 13	Least-squares minimum regression.	None, merely its use.	Reported actual effort = 22.1 + 2.44 * estimated peak staff ln(reported actual effort) = 1.61 + 0.662 * ln(estimated new KLOC) ln(reported actual effort) = 1.14 + 0.579 * ln(estimated total KLOC)	Estimated peak staff, reported actual effort hours in thousands. Estimated new KLOC, reported actual effort hours in thousands. Estimated total KLOC, reported actual effort hours in thousands.	Not specified.	30 projects from DoD developers at CMMI level 5.	Pearson and Spearman correlation, Mean magnitude of relative error (MMRE) and Prediction (PRED 25) accuracy.	With real effort values.	Second equation performs better than the third. However, the third one is useful for comparison to published literature.	No.	–
2015. 14	NHPP growth curve models (exponential, delayed S-shaped), Moranda geometric Poisson model.	They tested models and variables to discover the best ones to predict reliability.	Inputs: Implemented development size, number of test cases executed. Outputs: Cumulative number of detected faults, reliability.	Implemented development size, number of executed test cases, cumulative number of detected faults.	Agile.	Five datasets of actual agile software development projects from three development genres.	Mean-squared errors (MSE) and Akaike's information criterion (AIC).	With real effort values.	All reliability growth models presented good results.	No.	–
2014. 02	Rayleigh curve or distribution.	None, merely its use.	Inputs: Log of peak staff, log of ESLOC, log of production rate. Outputs: Log of defects.	Log of peak staff, log of ESLOC, log of production rate, log of defects.	Not specified.	Over 2,000 recently completed software projects from the QSM database.	None.	–	Such models, with multiple control charts, can be used in large projects, with multiple testing phases.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2014.12	Correlation, regression.	None, merely their use.	Inputs: Density of injected defects, percentage of defects removed / removal efficiency / defect correction effort by phase. Outputs: Injected / removed / escaped defects by phase, total effort by phase.	Density of injected defects, percentage of defects removed / removal efficiency / defect correction effort by phase, injected / removed / escaped defects by phase, total effort by phase.	Not specified.	They do not detail it.	None.	–	Project managers use predictive models to consolidate the process defined and to control the probability to achieve the objective.	Yes, but they do not detail it.	–
2014.24	System dynamic, control charts and capability calculations.	Their use in combination.	Inputs: Skills training factor, process training factor. Outputs: Software defects, project non-conformances, product size.	Skills training factor, process training factor, number of non-conformances, product size, total defects.	Not specified.	Data from 5 projects from a CMMI level 3 software factory with about 140 developers.	Student's t-test. They presented the model to a panel of three experts.	They designed 3 scenarios with the same values for independent variables.	Investment in process training results in a process with less variation and fewer defects.	No.	–
2012.05	Bayesian belief networks.	None, merely its use.	Inputs: Defect complexity, experience of the engineers, analysis effort. Outputs: Bug-fix effort.	Effort spent on bug fix, bug complexity, experience of engineers, analysis effort.	Not specified.	Dataset from telematics area in Automotive domain.	Accuracy and sensitivity.	With real values from the organization.	Accuracy is 75% and sensitivity is 67%. They performed what-if analysis regarding analysis effort and team experience.	No.	–
2012.16	Time-series cross-section regression analysis.	None, merely its use.	Total / major production defects in current month = $\alpha + \chi$ (total / major defects in previous month) + β1 (total / major unit-test defects in current month) + β2 (total / major unit-test defects in previous month) + β3 (total / major system defects in current month) + β4 (total / major system defects in previous month) + β5 (total / major regression defects in current month) + β6 (total / major regression defects in previous + ε	Number of (total / major) defects per month, number of (total / major) (unit test / system / regression) defects per month.	Not specified.	Reported production defects and maintenance activity logs produced by a CMMI-DEV level 3 organization.	The model was applied across six simultaneous maintenance projects. Various statistical tests (F test and Lagrange Multiplier test) and a root cause analysis of reported problems.	–	This model can serve as a reliable tool for predicting temporal patterns of production defects across multiple projects.	Yes, in six maintenance projects.	The resulting causal analysis and resolution action plan illustrate the value of the model results as inputs to organizational process improvement efforts.
2011.01	Artificial neural networks.	None, merely their use.	Inputs: Effort to reproduce, knowledge level of the developer, code complexity, changes to design, dependency on other modules, testing effort, impact on code base. Outputs: Total effort required for a bug fix.	Effort to reproduce, knowledge level of the developer, code complexity, changes to design, dependency on other modules, testing effort, impact on code base.	Not specified.	Dataset from telematics area of Automotive domain.	They computed the fit of the model to the data and the correctness of the prediction in percent.	With real values from the organization.	The model with dataset C yielded an accuracy of 70%.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2011. 20	NHPP reliability growth models, Maximum likelihood, Newton Raphson, Mean Value Control chart.	Imperfect debugging models where new bugs can be added while removing others.	Inputs: Time between failure Outputs: Mean Value chart with the differences between cumulative failure data.	Time between failure.	Not specified.	They used historical data to test the approach: 30 points.	They used 30 historical time between failure points to test the approach.	With Xie et al. (2002) control chart (time control chart).	The proposed Mean Value Chart detects out of control situation at an earlier instant than the situation in time control chart.	No.	–
2010. 02	COCOMO II, CMMI.	COCOMO II still relies on SW-CMM to assess its PMAT scale factor.	$PM = a * size^{E} * \prod_{i=1}^{17} EMi$ $E = b + 0.01 * \sum_{j=1}^{5} SFj$, with new values for PMAT scale factor.	Effort, size.	Not specified.	40 datasets from CMMI levels 1 to level 4, with 8 data points each level.	Relative Error (RE), Magnitude of Relative Error (MRE), and PRED (30%).	With COCOMO II and actual effort values.	New PMAT estimated effort closer to the actual effort than generic COCOMO II estimations.	No.	–
2010. 03	NHPP growth curve models (exponential, delayed S-shaped and inflection S-shaped).	They tested the models to identify the best one to predict code churn.	Inputs: Code churn history. Outputs: Cumulative code churn estimate.	Code churn (includes code addition, deletion and modification).	Open-source development.	12 packages included in Eclipse.	They performed experiments to predict code churn with the inflection S-shaped model. Mean magnitude of relative error (MMRE).	They compared the 3 NHPP model types to the actual values.	Inflection S-shaped model showed better fit to the real code churn in Eclipse than other models.	No.	–
2010. 09	Six Sigma DMAIC, multiple linear regression.	None, merely their use.	Defect density in systemic tests = 1.8955 - 0.5087 * percentage of defects in technical revisions - 1.6020 * unit-test coverage General project productivity = 32.087 - 3.637 * defect density in systemic tests + 11.71 * level of the requirements instability - 9.451 * level of continuous integration utilization - 0.8187 * level of experience * development environment	Percentage of defects in technical revisions, unit-test coverage, defect density in systemic tests. Defect density in systemic tests, level of requirements instability, level of continuous integration utilization, level of experience, development environment, general project productivity.	Not specified.	Data from the company.	They tested the model on five projects of the organization to verify its efficacy in predicting final productivity.	With the percentage difference between the planned productivity at the beginning of the project and at the end.	The model is a good way to obtain more precise estimations, to improve client satisfaction and to reduce variability of timeline, cost and productivity.	Yes, in 5 projects at a CMMI level 3 company.	Productivity estimation using the models is significantly more precise than traditional techniques such as, for example, the use of organization historical average.
2010. 14	Linear, inverse, quadratic, cubic and power regression analysis.	They tested regression analysis methods to identify the best one to predict inspection effectiveness.	Inputs: Code inspection rate. Outputs: Defect density.	Code inspection rate, inspection effectiveness.	Not specified.	Data from 45 code inspections performed by developers on 3 projects.	R-square, Fisher test and ANOVA. They used the model to improve the process and performed 39 inspections.	With real values. The pilot results were compared to the previous ones.	The study is useful to understand the impact of review rate on process performance.	Yes, in 39 inspections performed by 3 reviewers in 4 projects.	The average review rate changed from 800 LOC / hour to 215 LOC / hour with a standard deviation of 46. The average value for defect density improved significantly.
2009. 07	Multiple regression analysis, F test.	They use multiple regression analysis to specify parameters and F test to evaluate them.	% fixing effort = A * % defects injected during requirements + B * % defects injected during design + C * % defects injected during coding + D	Defect injection rates of requirements, design, coding, and testing activities, % fixing effort.	Iterative development.	Company data.	After applying the method for several years, they conducted interviews and analyzed historical data.	With actual project values and organization historical data.	Benefits observed: better management of process data and quantitative control of projects.	Yes.	Fixing models were coherent and covered the entire development lifecycle.

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2009. 17	Regression analysis.	None, merely its use.	Development process choice = α0 + α1 * (client specific knowledge) + α2 * (extent of client involvement) + α3 * (design and technology newness) + α4 * (estimated project effort) + α6 * (estimated code size) + ε1	Client-specific knowledge, extent of client involvement, design and technology newness, estimated project effort, estimated code size.	Offshore software development.	Data from 112 software project processes and performance from two different CMM and People CMM level 5 software companies.	The model's chi-square statistic value is significant, indicating that the model is statistically valid.	They used the model to find similar projects in group and compare their performance indicators.	Project managers could use this model, at the start of the project, to decide if changing some of the processes would result in better project performance.	No.	–
2009. 18	Estimation methods (COCOMO II, NASA and FP).	They attempted to experimentally verify the degree of validity of the estimation methods and to propose a correction factor.	Development effort in hours including software documentation (ED) = a * (product size in thousands of lines of code) ^ b * (adjustment factor for software documentation d) where, d = [1.01, 1.31]	Initial and final function points, estimated / real lines of code, estimated / real effort per development phase, documentation number of versions / estimated size / versions size / real size / (estimated / real) effort.	Not specified.	One experiment including development and documentation.	An experiment was developed and performed in a course in the 5th year study in Computer Science.	They compared their proportional value to the ones used in estimation methods.	Documentation effort fluctuated between 12.34% and 34.67% of total effort. COCOMO II documentation variable is within the calculated range.	No.	–
2008. 04	Intermediate COCOMO II, FP counting.	New values for some cost drivers adequate to projects of less than 10 PM (person-months).	$\text{Effort} = a * \text{KLOC}^b * \text{EAF}$, with new values for software development effort multipliers.	Effort, size.	Projects of size less than 10 PM.	Data from one project was used to derive new proposed values.	Data from two other projects was used to validate the values.	With Intermediate COCOMO II default multipliers.	This approach is useful for projects with size less than 10 PM. They had 30% improvement in effort variance.	No.	–
2008. 20	RATS risk system, multiple regression analysis, central limit theorem, non-parametric permutation.	They proposed transforming the continuous endpoint, such as profit, to a binary variable such as "Yore".	Inputs: (client reviews returned + specifications changed in the first month) / PM estimate, the sum of the scores for eight risk-related items and seven administrative items. Outputs: Risk failure of a project based on the profit rate (Yore = 1 when a project fails and Yore = 0 otherwise).	Productivity, profit rate, client reviews returned in the first month, specifications changed in the first month, person-month estimate, sum of the scores for eight risk-related items and seven administrative items.	Not specified.	48 data projects from RATS database.	Scatterplot of the cumulative estimated probability vs. the cumulative observed frequency of Yore = 1. Values agreed well, indicating the appropriateness of the model.	With real values.	To apply this, first estimate the risk of Yore for each project to identify those at higher risk. Then, perform any action to reduce risk and re-evaluate using data collected after the treatment.	No.	–
2008. 21	Rayleigh model.	None, merely its use.	(Defects at testing phases / all defects) = e ^ (-(positive constant) * (defects detected at phase / all the defects at phase or later or defect removal rate)) (Defects at testing phases / development size or defect density) = A * (defects at testing phases / all defects) + B	Number of defects detected in the phases, defect removal rate, development size, defect density.	Large projects with rigid waterfall development phases.	Performance data from 17 completed projects.	They performed linear regressions on project data. The second linear regression equation showed several outliers related to project characteristics.	With real values.	This model can be used to evaluate the effect of peer review, make review plan and set objectives for values for review activities, and evaluate the effect of each peer review activity.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2008.25	Petri nets.	None, merely their use.	Inputs: (Total defects, review / testing / rework time, failure / repair rate, MTTF, MTTR) per phase Outputs: Total defects, % reliability, % unreliability	Total defects rework time, failure / repair rate, MTTF (= 1 / failure rate), MTTR (= 1 / repair rate)	RUP-based development.	Reliability data from one module.	They simulated a Petri net with reliability data for each of the module cycles.	–	Results show number of defects being significantly reduced in incremental cycles.	No.	–
2008.29	Literature Seemingly unrelated regression technique.	Focus on a distributed development context subjected to the effects of work dispersion.	task dispersion = - 0.732 productivity task dispersion = - 1.525 quality learning investment = 0.630 productivity learning investment = 0.939 quality	Productivity, quality, process investments, task dispersion, integration intensity, learning investments, software size, team size, project management investment, up-front investment.	Offshore software development.	42 offshore software development projects of a CMM level-5 software organization.	Their built from data analisys.	–	Investments in structured processes and process-based learning routines mitigate the negative effects of work dispersion in offshore software development.	No.	–
2008.31	GQM, linear multiple regression.	None, merely their use.	Effort variance = -1.64 + 0.003895 size + 9.96 skill level	Size in unadjusted FP, skill level, effort variance.	Maintenance projects.	Project data collected over 5 years.	t-test and ANOVA test, Mean Magnitude of Relative Error (MRE), Median Magnitude of Relative Error (MdMRE) and PRED (0.25, 0.5).	With real values.	The model holds true for tasks having size between 100 UFPs and 3,877 UFPs.	Yes, to estimate 5 different change requests.	Variations in the predictions for the 5 change requests were used to further field-test the model as an empirical validation of the results.
2007.01	Linear regression, forward stepwise regression, two-stage least-squares.	None, merely their use.	ln(effort) = 4.49 + 0.61 * ln(size) ln(quality) = 1.38 + 0.3 * ln(size) ln(cycle time) = 4.23 + 0.27 * ln(size)	Size in KSLOC or FPs, effort in person-days, quality in number of defects, cycle time in number of calendar days	Not specified.	Data collected from 37 CMM level 5 projects at four organizations in software development outsourcing.	Statistical tests t, Shapiro-Wilk, White's, and Cook's distance. Magnitude of relative error (MMRE) percentage for N estimations.	With actual values.	High levels of process maturity reduce the effects of most factors in software effort.	No.	–
2007.09	Defect density curve, SWEEP software tool.	They applied a SWEEP tool used at a CMMI level 5 company on a hardware development project.	Inputs: Defect data. Outputs: Number of defects remaining in the software. The predictive model is the inverse of the defect density curve used by reliability engineers.	Number of defects.	Not specified.	Historical data from the same project only.	They used it in one project, with different defect sources to improve the confidence level of the results.	With real data.	The usage of SWEEP contributed to the success of the development effort in meeting its schedule commitments.	Yes, in a hardware development project.	Each set of defect data predicted that 50% of defects remained undetected.
2007.13	u control charts, design of experiments.	Their use together to model inspection process.	Inputs: The average cost of fixing defects in different stages, the cost of false alarm, mean shift when process goes out of control, amount of shift when the process goes out of control, control limits of defect density, the inspection module size. Outputs: The cost.	The average cost of fixing defects in different stages, the cost of false alarm, defect density, module size.	Not specified.	Historical data from the company, but they do not detail it.	None.	–	In most situations the optimum module size is between about 150 and 500 LOC, and as the process stability increases, the optimum module size increases.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2006. 22	CDM model (Cangussu et al., 2002).	A significant improvement over the previous version of the model. A new calibration algorithm for the CDM model.	Inputs: The workforce size, the average number of people working at a given time, the quality of the test process, the complexity of the software to be developed, the software type being developed, the defect detection constant of proportionality, the quality impact constant of proportionality, an estimate of the number of defects introduced into the software. Outputs: The resulting control values over actual and next checkpoints.	The number of defects detected/ eliminated per day.	Not specified.	–	They used a testing scenario at large software manufacturer. The test phase lasted 120 days over which a number of defects were discovered and removed.	–	The controller is capable of determining the appropriate changes required to drive the model to the specified desired behavior.	No.	–
2003. 05	SPC, ordinary multivariate least squares regression.	Their use.	Total effort of the work-packet = 0.12256 Number of software code components Total effort of the work-packet = 0.14109 Number of software code components + 8.925E-03 Number of candidate impacts Total effort of the work-packet = 2.45253 Sqrt(Number of software code components) + 7.02285 sqrt(Number of actual impacts) Total effort of the work-packet = 2.26257 Sqrt(Number of software code components) + 4.66005 sqrt(Number of non-standard actual impacts) + 4.66005 sqrt(Number of actual standard impacts)	Number of software code components, number of candidate impacts, number of actual impacts, number of actual standard impacts, number of non-standard actual impacts, number of test cases, actual effort measured as man-days, total number of employed maintainers, actual duration measured as number of calendar days.	Maintenance processes.	A Y2K remediation project for a large application portfolio composed of about 40,000 software components.	Using the leave-one-out cross validation approach they computed the mean relative error MRE and the following variants of the measure PRED: PRED 15 and PRED 50.	With real data.	The results of the regression models demonstrate a good repeatability and predictability of the effort required for a maintenance project.	No.	–
2003. 19	Outcome Based Control Limits, forward/ reverse	Their combination to create the models.	Inputs: Collected data. Outputs: Delivered defects to the customer. There are 5 formulas on paper.	Size of the project, total number of defects injected into the software, percentage of total defects injected at phase	Not specified.	Actual project data used for model parameters where possi-	An example where a software development firm is being contracted to	-	The bi-directional simulation models identify not only when the project is “Out of Control",	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
	simulation models.		Inputs: Collected data. Outputs: Number of defects allowed to escape as set by the Outcome Based Control Limits. There are 5 formulas on paper.	i, number of defects injected at phase i, number of defects that escape detection at phase i. percentage of latent defects in the code at phase i that are detected and corrected, number of defects that are detected and corrected at phase i, inspection or test effectiveness required to achieve the Outcome Based Control Limits at intermediate phase.		ble. Specially, actual defect injection and detection rates collected by the company.	do a 52 KLOC revision to an existing product. Process diagrams, model inputs, and model parameters were reviewed by members of the software engineering process group as well as senior developers and managers for their fidelity to the actual.		but also provide an indication of what magnitude of improvement needs to be made in order to bring the project back on track.		

Table B3. Automated process performance analysis methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2007.19	SPC, data mining, Modified Centered CUSUMS control charts.	They extend the idea of the Modified Centered CUSUMS, and propose a new data selection procedure so that the associative discovery technique can be used in retrospective SPC analysis.	Inputs: Process historical data. Outputs: Dataset for mining the causes of the mean change of the process, dataset for mining the cause of the variance change of the process, association rules in the form: "If C inputs are used then R will happen."	Any indicators.	Not specified.	–	They suggest using Leverage to indicate the validity and importance of the rules generated.	–	Data mining method can be used to find the hidden knowledge from the data, and to identify the causes of process failure or success for quality improvement. In addition, this information can be used for the cause-and-effect diagram in online process control.	No.	–

Table B4. Automated process performance analysis methods and tools: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2016.11	Their previous work.	An automated tool to support their complete approach.	Inputs: Performance model, data calibration file, type and input file with performance data to be analyzed. Outputs: Table view with detailed evaluation of all performance indicators (PIs) for all projects, Report view with an overall summary or project-by-project most relevant top-level performance problems and potential root causes, Indicator view with the behavior of each PI across the projects under analysis and associated model definition and calibration information, Diagram view with the same information as the Report view plus additional details.	It can work with any indicator, but it needs a model with the necessary information.	Not specified	–	Case study with seven projects performed by a PSP developer during the PSP Fundamentals and Advanced training.	Analysis performed by the tool was compared to that performed by the student.	The main advantage of this analysis is to point out problematic areas where subsequent manual analysis should focus.	No.	–
2012.06	A previous method for monitoring technical quality of software products.	A method to automatically detect events to improve the previous method's responsiveness and scalability.	Inputs: Periodical snapshots of the source code. Outputs: Indication of outstanding events, trends or other signs in the data that indicate potential problems, an alert email with this information.	LOC, duplicated LOC, McCabe complexity, dependency counts.	Not specified.	–	They used a random sample of 98 alerts from the evaluation period and interviewed the receivers of these alerts to determine their desirability and expectedness.	–	One potential issue introducing the alert service in a new environment is choosing the thresholds for sustained and abrupt event detection.	Yes, it was used over 20 months for a subset of systems.	Approximately 4% of generated data points were classified as events, causing 876 alerts.

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2012. 20	SPC, CUSUM control charts, CUSUM-Shewhart control charts	Using performance regression testing at beginning of the development cycle and automation of most of the work.	Inputs: Data collected from continuous building and automated tests. Outputs: Panels showing unusual performance issues and assisting in identifying their causes through data and differential profiler (comparisons between a previous profile and the abnormal one to find code portions which were modified).	Database performance variables.	Stable software evolution.	Historical test data on DBMS evolution.	Simulations to decide issues such as the number of points under analysis. Case study and experience of the organization when using the proposed framework.	With the manual procedure established previously.	They were able to migrate their weekly or monthly monitoring cycle of several key performance metrics to a daily cycle, with faster feedback to the developers, therefore reducing investigation cost.	Yes.	They were able to remove most of the manual overhead in detecting anomalies and reduce the analysis time for identifying the root causes by about 90 percent in most cases.
2010. 07	SPC, agents.	An approach to define and monitor software improvement goals.	Inputs: Strategic, tactical and project needs. Outputs: Steps to help to define and plan strategic, tactical and project goals, considering SPC and software measurement, alerts for real or potential deviations from related measures collected during project.	–	Not specified.	–	Use of strategic and tactical planning phases in academia with a survey. The infrastructure is under development.	–	The survey showed that professionals expected good benefits from the enactment of the strategic planning accomplished.	Partially.	Throughout strategic planning it was possible to identify issues that could threaten the achievement of the defined goals.

Table B5. Process performance analysis methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2017. 03	SPC, Fishbone analysis, correlation analysis, regression analysis, Monte Carlo Simulation and sensitivity analysis tools.	Their use on an organizational process asset library architecture to support high maturity.	Inputs: Business Objectives. Outputs: Process performance models and quality and process performance objectives.	Sales/ department revenue, labor utilization rate, delay rate, production efficiency, rate of requirement change, defects density in the acceptance.	Not specified.	-	One example of its application.	-	This architecture serves the high maturity process improvements conveniently and it is with the extensibility and easily implemented by software enterprises.	No.	–
2014. 12	SPC.	None, merely their use.	Inputs: Historical data. Outputs: Steps followed to analyze process performance including mean and standard deviation analysis, special causes of identified variation, demographic data analysis, data removal.	–	Not specified.	They do not detail it.	None.	–	The usage of the analysis of objectives approach helps the company during the process of achieving high maturity levels.	Yes, but they do not detail it.	–
2013. 08	GQM, SPC, EWMA control charts, XmR control charts.	They report the steps used.	Inputs: Historical data. Outputs: Steps followed to identify metrics related to organizational business objectives, determine if it was worth applying SPC to a given metric, choose the frequency that allows reacting as quickly as possible to process changes, choose control chart types, verify process stability, and use limits as baseline.	Percentage of defects rejected / found in operation, percentage of high-severity defects identified in production / testing, percentage of defects caused by faulty logic.	Not specified.	852 defects reported during 2011.	They conducted a SCAMPI type C assessment.	They compared results of EWMA charts and XmR charts.	XmR chart is the most useful, if it is complemented with additional stability tests. EWMA charts have narrower limits.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2012. 24	SPC, control charts.	They propose two prepro-cessing steps on the counter data before constructing the control chart.	Inputs: Process inputs such as the load, process outputs such as perfor-mance counters. Outputs: Steps involving running tests, generating test baselines, pre-processing counter data before con-structing control chart, creating con-trol chart, and detecting performance regression violations.	CPU utilization, MySQL IO read bytes/sec, Tomcat pool paged bytes, Tomcat IO data bytes/sec, Tomcat IO write bytes/sec, Tomcat IO data op-erations/sec, Tomcat IO write op-erations/sec.	Not speci-fied.	Historical data from the com-pany.	Two case studies, one on a large en-terprise software system and the other on an open-source software sys-tem.	With engi-neers' classi-fication in case study one, and with five injected programming scenarios in case study two.	They identified test runs with performance regressions having 75% precision and 100% recall in the first case study. They could identify four out of five injected common inefficient pro-gramming scenarios in the second case study.	No.	–
2012. 27	CMM, CMMI, PSP, Balanced Score-card (BSC) Team Software Process (TSP).	They report their evolution and results.	Inputs: Historical data. Outputs: Steps followed to analyze process performance as they adopted CMM, CMMI, PSP, BSC and TSP, in-cluding control chart examples.	Schedule, effort, ins-pections, defects.	Not speci-fied.	–	Examples.	–	Average schedule / ef-fort deviations im-proved, system test du-ration was reduced, team inspections and personal reviews re-moved more defects, profit averaged, test and rework reduced.	Yes, but with no detail.	They can field larger team sizes and main-tain schedule, cost, and quality within known process capabil-ity.
2011. 14	Literature reports.	They create a catalog with measures and indicators for high maturity.	Inputs: Process performance analysis issues. Outputs: A catalog of measures and indicators related to the processes for SPC.	–	Not speci-fied.	–	Its use at a CMMI level 3 organization.	–	The application of the catalog proved to be simple and clear to the project managers.	Yes.	They analyzed time, cost and product quality, and derived some improve-ments.
2011. 17	SPC, u-charts.	None, merely their use.	Inputs: Historical detected defects. Outputs: A quality system with u-chart of defect density per program, distri-bution of defect density to allow cal-culation of probabilities.	Defect density.	Not speci-fied.	Organization historical data.	They tested it in seven system devel-opment projects.	With real de-fects values.	Data collected in the u-chart can be used as basic data to predict remaining defects.	Yes, in seven projects.	Prediction re-sults were con-sistent with the actual results in five of the pro-jects.
2011. 23	Design of experi-ments.	They provide one example.	Inputs: Desired and actual values of cumulative failures. Outputs: Steps to use statistical tests to estimate whether there is a signifi-cant difference between desired and actual values.	Defects over time.	Not speci-fied.	Shuttle failure data from NASA.	They used the method with histori-cal data.	–	There was not a statis-tically significant differ-ence between cumula-tive failures and target values.	No.	–
2010. 12	CMMI, visual pro-cess modeling lan-guage (VPML), an-alytic hierarchy process.	A method that supports high-maturity pro-cess improve-ment.	Inputs: Historical data. Outputs: Steps for process analysis including VPML-based process mod-eling, automated process simulation, process evaluation, rule-based pro-cess optimization, identification of op-timized process priority.	Duration, cost, qua-lity.	Not speci-fied.	–	They summarized how SPs in each PA of CMMI level 4 and level 4 can be sup-ported by the method.	–	Practices show that this method greatly eases the implementa-tion of high-maturity process improvements.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2010. 21	CMM, Continuous Quality Improvement (CQI), SPC.	Their combination.	Inputs: Historical data. Outputs: Steps followed including putting together an improvement team, defining a clear aim, identifying and defining measures of success, brainstorming potential change strategies to produce improvements, planning, collecting, and using data for facilitating effective decision-making, applying SPC.	Change Management process: emergency tickets per week, defective items.	Not specified.	Organizational data in IT services.	One example of its application at a real company with improvements.	–	The approach can serve as the basis for a roadmap to enable IT organizations to incrementally "move up" the CMM capability chain, and to eventually achieve world-class operational results.	Yes.	Average weekly defect ticket rate dropped, and process variability improved.
2010. 29	SPC, c-charts.	They identified six common quality evolution patterns.	Inputs: c-charts with historical defects data. Outputs: Six quality evolution patterns for SPC, with expected effects and possible actions.	Defects.	Software maintenance and evolution.	Defects from Eclipse and Gnome projects.	Various examples using data.	–	c-charts and patterns can help QA teams better monitor and understand quality evolution and prioritize QA efforts.	No.	–
2009. 02	SPC, experience from empirical investigations in industrial contexts.	They propose four “monitoring problem – SPC-based solution” patterns.	Inputs: Historical data. Outputs: Four “monitoring problem – SPC-based solution” patterns that assist in defining baselines, detecting anomalies, investigating root causes and performing tuning actions.	Number of lines of code / man-hours spent for reverse engineering or restoration of a Cobol program.	Not specified.	Each pattern was obtained from the generalization of their experience in industrial contexts.	One example applied to a legacy dataset. Project data was used to validate whether SPC would have been able to point out the known process performance changes.	–	They were able to characterize the process in use, continuously tune monitoring sensitivity and identify all the known improvements.	No.	–
2009. 07	Multiple regression analysis, F test.	They identify appropriate control points in each iteration.	Inputs: Defects data for each defect activity. Outputs: Steps to estimate defects removed in each iteration, analyze defects detected and corrected before integration testing of each iteration to predict defects removed in integration testing (same steps apply after all iterations with system testing), and estimate defect-detecting / defect-fixing effort of system testing.	Defect injection rate / defect removal effectiveness per phase, pre-release / post-release defect density, productivity, defect injection distribution, % detecting effort, % fixing effort, test / rework efficiency.	Iterative development.	Company data.	After applying the method for several years, they conducted interviews and analyzed historical data.	With actual project values and with organization historical data.	Benefits observed: better management of process data and quantitative control of projects.	Yes.	Effort, schedule, and defects were estimated based on project objectives and organization baselines. Schedule, cost, and quality were monitored.
2009. 12	Software reliability models.	They created a framework for weapons systems.	Inputs: Historical data. Outputs: A framework with stages and activities regarding domain analysis, establishment of software reliability goals, data collection, data analysis, evaluation of software reliability, application of improvements.	List of metrics for requirements tracking, complexity, inspection and review, testing and reliability growth models.	Not specified.	–	They used the framework on a company’s historical data on two products.	–	The weapons system with the framework applied showed improvements in software quality.	Yes.	They derived some process improvements.
2009. 26	Their previous work, data envelopment analysis, analytical hierarchy process.	They added decision-making preferences, and scaled project assessment.	Inputs: Data from PSP metrics. Outputs: Capability assessment results of the PSPs, Returns to Scale analysis.	Schedule, scale, scale / time estimation accuracy, reciprocal of defect density, process yield.	PSP.	–	They performed an experiment on a standard and representative PSP dataset selected from Putz’s book.	They calculated three different capability scores.	Incorporating decision-making preferences can ensure the assessment results will be consistent with the organizational objectives.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2009.28	SPC, GQ(I)M, IDEAL.	They create a model to perform SPC using GQ(I)M.	Inputs: Historical data. Outputs: A model with steps for defining indicators for organization business goals, choosing project process subprocesses, collecting metrics, mapping metrics to subprocesses, evaluating contribution of subprocesses, controlling and improving subprocess performance.	Project tracing and managing subprocess: schedule / cost / workload / code / file deviation rate, code / file defect density, defect injection rate.	Not specified.	Data from the nine software projects.	One example.	–	They merely explain the method and the example, but do not provide any conclusions.	No.	–
2008.07	SPC, chi-square, regression, ANOVA.	None, they merely report their use.	Inputs: Historical data. Outputs: Steps and tools to support establishing goals, collecting data, storing data, providing data, analyzing data, performing causal analysis and decision-making, and taking corrective action.	Product sizes, inspection rates, preparation rates, defect density.	Not specified.	–	Merely their experience.	–	The authors describe challenges in deploying statistical methods, challenges in implementation, and various guidelines.	Yes.	Variability of unit cost and average cost declined, post-delivery defect density reduced.
2008.09	SPC, Hotelling's T2 charts, multivariate cumulative sum control charts, multiple linear regression, ANOVA, t-test, PSM.	They propose a multivariate analysis instead of a univariate analysis.	Inputs: Historical data. Outputs: Steps of multivariate analysis to identify causes of out-of-control signals.	Defect density, defect rate, component complexity, requirements change, wrong implementation, product size, used effort.	Not specified.	–	One example of use with data from a real project. The project comprised 7 work packages divided into 22 scheduled tasks.	–	They show an application of the method using real data, but do not derive any conclusions.	No.	–
2007.15	SPC, ISO/IEC 15393.	They report common errors based on a review of a data analysis process.	Inputs: Historical data. Outputs: Common errors in process analysis regarding non-normal data distributions, data aggregation, and misleading metrics.	Productivity.	Not specified.	Historical data from 1,093 enhancement projects over four years.	Examples.	–	Lessons learned from analyzing the data into three do's and four don'ts for productivity and measurement analysis.	No.	–
2007.26	GQM, Balanced Scorecard, 5W1H.	None, they merely report their use.	Inputs: Historical data. Outputs: Steps followed to identify organization information needs, establish goals, questions and metrics and related processes, review indicators, choose statistical tools, analyze processes, and generate baselines.	Degree of client satisfaction, on-time delivery, cost deviation, productivity, defect containment rate.	Not specified.	Historical data since CMMI level 3.	–	–	The tool set selected proved to be helpful to attain CMMI level 4 requirements, although it requires a significant amount of analysis effort.	No.	–
2006.03	SPC.	They monitor the execution of a process by measuring the supporting ones it depends from.	Inputs: Data collected and classified in problem categories. Outputs: Limits that are used to monitor the process performances, and are automatically each time the process starts to become unstable.	Problems detected from each of the user sites.	Heterogeneous and geographically distributed sites.	-	None.	-	This allows analyses from more a general to a more specific level of detail. Such analyses allow to make previsions on the process being monitored and controlled."	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2006.05	SPC.	A framework that interprets SPC and applies it from a software process point of view.	Inputs: Historical data. Outputs: Process performance monitored and action taken.	Productivity (LOC/hours).	Not specified.	-	The framework has monitored a reengineering process of an aged banking application that needed to be rejuvenated.	-	SPC-Framework appears as an effective and nonintrusive instruments for supporting the project manager during project execution.	Yes.	SPC monitoring highlighted all relevant shift occurred in the process performance, and suggested when to recalculate the reference to be more/less accurate.
2006.28	SPC, Data mining, QSM's Software Lifecycle Management (SLIM) tool.	Using them together.	Inputs: Historical data on some tools. Outputs: Process control and improvements identifications.	Effort, cost of quality, cost of poor quality, inspection effectiveness, size defect rate.	Model-driven development.	-	Its use.	-	MSR method was effective in helping manage these MDD projects.	Yes.	Using SLIM-Control for planning is an iterative process. At milestone points, they added new data into SLIM for replanning.
2005.11	SPC, Shewhart control charts, Select Cause Control (SCC) charts	They try to help when the variation probably results from other processes.	Inputs: Historica data, type of process quality intented. Outputs: Scenario 4 (Shewhart charts of upper process is out of control,and Shewhart charts and SCC charts of target process are under control) or scenario 8 (Shewhart charts of upper process,and Shewhart charts and SCC charts of target process are under control).	-	Not specified.	-	None.	-	None.	No.	–
2004.02	CMM, QFD, Goal-driven Measurement Process, GQM, Simplified Quality Functional Deployment, ISO/IEC 9126, Simplified House of Quality.	Their use.	Inputs: Business goals. Outputs: Project measure analysis and diagnosis.	Five areas: size, effort, schedule, defect, and risk. Plus project staffing, duration of activities, and changes.	Software maintenance.	-	Their use on four pilot projects.	They compared level 4 metrics with previous level 3 ones.	The cultural challenges imposed by the SPC implementation must not be underestimated. Project leaders must be able to think about measures in quite a different way.	Yes.	A 5% increase in costs on each project can be expected owing to SPI activities.
2004.04	Six Sigma, TQM, SPC.	None.	Inputs: Historical data. Outputs: Controls, like mail notifications, that take advantage of the improvement zone between 3σ and 6σ process performance.	Back orders (percentage of orders).	Not specified.	-	Examples.	-	Organizations that can achieve such tight performance in key design dimensions can yield enormous benefits.	No.	–
2003.02	SPC, Feedback Process Control, SPC log.	Their combination.	Inputs: Business goals. Outputs: Process changes when necessary.	Software Test Process(STP).	Not specified.	-	Simulation. Data is generated by randomly slowing down the exponential decay of errors for the process.	-	The use of a control mechanism decreases the dependency of the process on the manager skills and thus improves its controllability.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2003. 03	SPC.	A variant of SPC based on a logarithmic transformation.	Inputs: The expected decay (Target Center Line) and an initial value should be provided. Outputs: A process with an exponential behavior controlled with SPC.	The number of errors found per time unit.	Process presenting an exponential behavior.	-	The SPClog technique presented is evaluated using simulation and data collected from a commercial project.	-	Results demonstrate the applicability of the approach, and its correctness, when applied to the testing phase.	No.	–
2003. 07	SPC, control charts.	Some tips on its adoption for software.	Inputs: Collected data on inspection. Outputs: A better interpretation of the control chart, and the identification of problems even when the defects are inside control limits.	Size of the product or lines of code inspected, rate of preparation in lines of code per hour, number of staff hours expended, number of people on the inspection team, initial number of defects present in the code.	Not specified.	-	None.	-	SPC is a powerful tool to optimize the amount of information a manager must use to make actionable decisions about the process. However, to avoid drawing incorrect conclusions from SPC charts, we must consider multiple measures.	No.	–
2003. 11	Six Sigma	The suggestion of their use on software engineering.	Inputs: Project initiating. Outputs: Software developed with less defects and better customer satisfation.	Defects per million opportunities (number of keystrokes, number of lines of non-commented source code, number of function points, and number of executions).	Not specified.	-	None.	-	Although 6SSP is still relatively premature with overwhelming unresolved issues, it offers hope to those who are just about to resign to the "late and buggy" work of the software world.	No.	–
2003. 12	SPC, XmR charts, u-charts, cause-and-effect diagram.	Their use.	Inputs: Historical data. Outputs: Process improvement opportunities.	Coding-and code-review scenario: average time spent for preparation for review, preparation speed, review speed, defects detected in the code review of unit, defects detected in unit testing, module testing, and system testing.	Not specified.	-	They tested the method on two process automation projects (Projects 1 and 2) and two consumer electronics projects (Projects 3 and 4).	-	By managing the code review process, they can make it conform to specifications. By controlling the process, they can maintain it within its control limits. By improving the process, they can improve its capability. The chart itself does not tell what to change; it only identifies the improvement opportunities.	Yes, on 4 projects.	They performed some process improvements throught the process analysis.

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2003.16	Six Sigma, DMAIC.	The model mapped software development life cycle against six-sigma quality cycle and project management life cycle.	Inputs: A project starting. Outputs: Improvements done, evaluated and shared.	The area of critical mass was determined to be unit testing.	Not specified.	-	The use by an IT company, a supplier of medical equipment, facing the Internet competition challenge.	-	Six-sigma quality can push the bar of operational excellence to a level of perfection never before imagined.	Yes.	Not only did the client benefit, but also the IT organization: reliability and accuracy increased by 88%, customer complaints were reduced by 75%, cycle time reduction of 28%, and rework was reduced by 82%.
2003.17	SW-CMM, Six Sigma, Pareto analysis, cause-and-effect analysis, Quality Functional Deployment.	Six Sigma helps to match the process improvement goals with customer expectations and to predict and measure the capability in schedule, effort, and quality.	Inputs: - Customer needs. Outputs: - Process performance managed and continuous improvement.	Rework index, failure cost, schedule slippage.	Not specified.	-	Their use on one center.	-	Six Sigma and SW-CMM complement each other and together can help an organization meet its process improvement goals.	Yes.	They reduced its in-process failure cost from 5 to 1 percent, thus reducing the cost of quality. The center also had process and technology improvements for cycle-time reduction and productivity improvement.

Table B6. Process performance analysis methods and tools: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2014.29	SPC.	A software tool with knowledge to guide its application.	Inputs: Historical process data. Outputs: Tool support with steps to prepare for performance analysis, verify stability, verify capacity, establish performance models, monitor stability, monitor capacity.	Coding effort per requirement.	Not specified.	-	One example. A survey was conducted with specialists to evaluate the proposed set of activities and tasks.	–	None. It is an ongoing work and authors point out future work and evaluations.	No.	–
2012.14	SPC, difficulties in applying SPC, previous tool.	A process to help with identified difficulties and an improvement tool to support this process.	Inputs: Organizational needs. Outputs: Tool support to perform measurement planning for SPC, measurement collecting, measurement results analysis and process establishment and control.	Not specified.	Not specified.	-	They analyzed process adherence to the MR-MPS maturity model and provided tool screens.	–	They believe that integrating these features will reduce inconsistencies and make SPC techniques less costly for organizations.	No.	–
2012.30	SPC, Six Sigma.	They verify requirements for quantitative management and specify software process components and measures.	Inputs: Historical process data. Outputs: Tool support to capture data, assess the suitability of a software process and measures for quantitative analysis, analyze a software process with respect to its qualifying measures using SPC.	They applied in 12 different processes.	Not specified.	-	They applied A2QPM retrospectively to 12 processes at six different organizations.	–	The authors provide a list of lessons learned from this experience.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2007.14	SPC.	They build a tool to support SPC.	Inputs: Historical process data. Outputs: Tool support to import data from other tools, organize data, define metrics and choose ones appropriate for SPC, choose correct SPC techniques, guide rational sampling, define derived metrics, interpret chart outcomes, guide what-if analysis based on rational sampling.	Bug creation / actual finish / estimated finish date, aging / estimated aging, estimation variance / capability, priority, problem source, status.	Emergent and low maturity organizations.	-	They analyzed all the bugs reported during 6 months (62 data points).	–	With a tool which guides users for rational sampling and metric utilization, an organization can apply SPC techniques and attain the ability to understand its processes based on quantitative data.	No.	–
2005.14	SPC GQM PDCA	The creation of the model and supporting tool.	Inputs: Historical data, process goals. Outputs: Quantitative objectives controlled by measures focused on the stable control of these objectives. They use x − S chart to measure and control the productivity of the process.	Earned value, frequency of process used/changing/tailoring, defect ratio, effort distribution, schedule variance, customer/ stakeholder satisfaction, stability of defect ratio/productivity/schedule variance.	No.	-	Examples of its application through maturity levels 2 to 5.	-	AMM has been adopted and implemented in SoftPM. It can support software organizations manage their process effectively under the appropriate measurement.	No.	-

Table B7. Process performance baseline building methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2015.10	Data Envelopment Analysis with Variable Returns to Scale (DEA VRS) model.	They use DEA VRS to analyze productivity. They analyze effort separated into three types.	Inputs: Historical data with inputs, outputs and decision-making units (DMUs). Outputs: Frontier with best-performing DMUs.	Project size, development / quality conformance / software maintenance non-conformance effort.	Not specified.	79 software development projects.	They applied statistical tests and interviewed people about the highest- and lowest-productivity products.	They compared different ways of using the same technique, and interviewed people to confirm their results.	DEA can be used to conduct efficiency analysis and adopt good practices from frontier projects and avoid pitfalls of non-frontier projects.	No.	–
2014.12	SPC.	None, merely their use.	Inputs: Historical data. Outputs: Steps followed to consolidate baselines.	Not specified.	Not specified.	–	None.	–	The usage of the baseline consolidation approach helps the company to achieve high maturity levels.	Yes, but they do not detail it.	–
2012.15	SPC, XmR control charts, grand mean.	They report steps used to generate baselines.	Inputs: Historical data. Outputs: Steps followed to select measures, control charts, define baselines, split baselines, and maintain baselines.	Defects per 1,000 lines of source code in requirement code reviews.	Not specified.	–	One example of an XmR chart.	–	The organization realizes benefits both before projects begin and while projects are executing.	Yes, but they do not detail it.	Grand means are useful and give the project team an overall sense of what the defect rate is. They use this to estimate rework effort.
2009.07	SPC, XmR charts.	They suggest a list of measures for baselines.	Inputs: Historical data on 10 suggested metrics. Outputs: XmR charts of the metrics.	Defect injection rate, defect removal effectiveness, prerelease defect density, test efficiency, and rework efficiency.	Iterative development.	Company data.	After applying the method for several years, they conducted interviews and analyzed historical data.	With actual project values and with organization historical data.	Benefits observed: better management of process data and quantitative control of projects.	Yes.	Effort, schedule, and defects were estimated based on project objectives and organization baselines.
2008.03	XmR charts, Six Sigma.	None, merely their use.	Inputs: Data on turn-around time for severity 1 incidents. Outputs: XmR chart of turn-around time.	Incidents per severity, turn-around time for severity 1.	Application Service Maintenance.	Incident data from nine months of a project.	Just one example of its use.	–	XmR charts can be used to create baseline values for turn-around time to set the control limits. This helps in monitoring the project using SPC.	Yes.	They achieved 50 percent reduction in average turn-around time for incidents.

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2006.27	SPC, GQM, PSM, Pareto, causal-and-effect diagram, scatter chart.	The creation of the method.	Inputs: Business goals of the organization. Outputs: Process performance refined continuously.	Coding process and requirement management process: relative schedule variation of task, module defect density, requirement change rate.	Not specified.	-	BSR was applied on 3 organizations throughout their process improvement lifecycle from lower level to higher level.	-	BSR is providing an effective method to establish and maintain the process performance baseline when the organizations want to manage these processes quantitatively. BSR method is also helpful for establishing process benchmark for software industry.	Yes.	The experiences of the three organizations validate that BSR approach is effective for evolving, establishing and refining process performance baseline.
2004.02	CMM, QFD, Goal-driven Measurement Process, GQM, Simplified Quality Functional Deployment, ISO/IEC 9126, Simplified House of Quality.	Their use.	Inputs: Data to be collected. Outputs: Process capability baseline.	Five areas: size, effort, schedule, defect, and risk. Plus project staffing, duration of activities, and changes.	Software maintenance.	-	Their use on four pilot projects.	They compared level 4 metrics with previous level 3 ones.	The cultural challenges imposed by the SPC implementation must not be underestimated. Project leaders must be able to think about measures in quite a different way.	Yes.	A 5% increase in costs on each project can be expected owing to SPI activities.
2003.18	SPC, GQM, run charts, XmR control charts, frequency histograms, box plots, u charts, zone charts.	Their use.	Inputs: Historical data on inspection process. Outputs: - Baselines for the inspection metrics. Inputs: Historical data on inspection process. Outputs: - Baselines for the inspection metrics.	Inspection rate (LOC/hr). Error Density (Errors/KLOC).	Not specified.	A total of 165 observations were available about inspection process.	Their application to establish baselines for the inspection metrics.	-	Software inspections help eliminate the chaotic impact of software defects encountered during testing and field operations. Software inspections supply important measurements and metrics.	No.	-

Table B8. Control chart specifications: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2016.17	NHPP Logarithmic Poisson execution time model.	They applied it to defects found in issue tracking systems for open-source product development.	Inputs: Data on defects in issue tracking system over time. Outputs: Control chart of fault-count data per development progress.	Number of defects in issue tracking systems over time.	Open-source software (OSS).	Data from Android and Thunderbird open-source issue tracking systems.	One example.	–	They can judge statistical stability and estimate additional development time for attaining failure intensity.	No.	–
2015.14	NHPP growth curve models (exponential, delayed S-shaped), Moranda geometric Poisson model, u control charts.	They propose a control chart to statistically assess agile software projects.	Inputs: Data of detected faults. Outputs: u control chart of the number of detected faults per development size per iteration number. Inputs: Data of detected faults. Outputs: u control chart of the number of detected faults per test-case per iteration number.	Implemented development size, number of executed test cases, cumulative number of detected faults.	Agile.	Five datasets of actual agile software development projects from three development genres.	Examples with real data.	–	They can quantitatively assess the quality of software products developed using u control charts with test cases or development size metrics.	No.	–
2014.01	u control charts.	One example of its use.	Inputs: Data of defects count and functional size for code peer review process. Outputs: u control chart of defect density per code peer review.	Code peer review process, defect density, functional size, defects count.	Not specified.	Nine samples or modules which were completed in 2008 with data on peer reviews.	One example.	–	They show one example with outlier identification and treatment.	It appears so, but they do not explore this.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2014.02	XmR control charts.	One example of its use.	Inputs: Data of defects. Outputs: XmR control chart of new defects per week. Inputs: Data of defects. Outputs: XmR control chart of the ratio of defects discovered to defects resolved per week.	New defects per week, defects discovered / defects resolved per week.	Not specified.	–	One example.	–	Control charts can be used to determine whether apparent changes in defect rates are significant.	No.	–
2014.08	EWMA Q control charts.	They test that control chart.	Inputs: Historical data. Outputs: EWMA Q control chart of the data, modified Fast Initial Response EWMA Q control chart of the data.	Number of defects detected in a short-run test phase.	Not specified.	–	Mean of the one-step-ahead forecast errors, mean absolute scaled error, accuracy measure, smoothed error tracking signal.	With average naïve forecasts.	EWMA Q control charts are attractive to control and monitor software development process.	No.	–
2014.21	SPC, order statistics, Pareto Type II distribution, half-logistic distribution, control charts.	They test both control charts on software reliability.	Inputs: Software failure data. Outputs: Control chart of half-logistic distribution of failure data. Inputs: Software failure data. Outputs: Control chart of Pareto Type II failure data.	Fault ID, time of fault.	Not specified.	Software failure data reported by Musa with 69 faults.	They generate failure control charts for 5th-order statistics for both models.	They compared both control charts.	Failures are detected at early stages with Pareto Type II model then with half-logistic distribution.	No.	–
2013.04	Q charts.	They test Q chart in software process.	Inputs: Small samples and short runs historical data. Outputs: Q control charts of data.	Code inspection rate, code complexity, defects per type identified during inspection, SPI, CPI.	No.	In one of the examples, the Q chart was used in a software project from a CMMI level 4 organization	Three examples.	With conventional XmR control charts.	Q chart allows monitoring process performance using a small amount of data in early development stages.	No.	–
2013.12	NHPP Burr distribution model.	They used the model on software reliability.	Inputs: Failure data. Outputs: Control chart of a function of the time between failures observations.	Failure time, failure interval.	Not specified.	31 failure data, but not detailed as real data.	Laplace trend test. Results show it is possible to estimate reliability.	–	Mean Value Chart detects out-of-control situations at an earlier instant than time control chart.	No.	–
2011.23	Statistical quality control, SPC. SPC, Poisson control charts.	They provide one example.	Inputs: Failure data. Outputs: Control chart of failure counts per test. Inputs: Failure data. Outputs: Poisson control chart of failure counts per test.	Defects over time.	Not specified.	Shuttle failure data from NASA.	They used the method with historical data.	–	They identified an out-of-control situation.	No.	–
2010.26	SPC, u-charts.	They provide one example.	Inputs: Defect data. Outputs: u control charts of defect density per priority level per document type per implementation / maintenance. Inputs: Inspection data. Outputs: u control charts of inspection performance per inspection type per document type. Inputs: Rework data. Outputs: u control charts of rework percentage for type (code / document).	Defect density, inspection performance, rework percentage.	Not specified.	Trouble reports from requirement documents and design documents from seven projects.	They investigated out-of-control points to find an interpretation for them.	Three-sigma limits results were compared to 2 standard deviation limits results.	Control charts are efficient in monitoring process stability and can be used by lower-level software industries.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2009. 09	SPC, CMMI, Scrum, Lean.	Putting them together.	Inputs: Build data from servers. Outputs: Control charts of fix time after failed builds. Inputs: Stories effort data from standard checklist. Outputs: Control charts of flow in stories implementation.	Fix time after failed builds, flow in stories implementation.	Scrum.	Data from projects of the company over the years.	They used in organizational data.	They compared projects productivity and found two with better performance.	Using CMMI and Scrum together results in significantly improved performance while maintaining CMMI compliance.	Yes, in two projects.	They made improvements based on two projects with better performance.
2009. 11	Moving average model, bootstrap scheme.	They propose a control chart to help decide on release time.	Inputs: Defect data. Outputs: 3-term moving average control chart of the number of remaining software faults per time index evaluation.	Number of remaining software faults.	Not specified.	–	One example.	–	This is a solution to make the right judgment on software release time, and provide quality monitoring.	No.	–
2009. 27	SPC.	They propose a transformation on data before using SPC.	Inputs: Minimal / maximal / expected time estimates to accomplish tasks, time spent on tasks. Outputs: XmR control charts of the transformed values.	Minimal / maximal / expected time estimates to accomplish tasks, time spent on tasks.	Not specified.	–	One example.	–	Characteristic values of software process can be unified and SPC can be applied to monitor running software processes.	No.	–
2006. 09	MiniTab, MS Excel, SPC, XmR control charts.	Just their use.	Inputs: Data from the Integrated Master Schedule (IMS) to track schedule progress taken against plan. Outputs: XmR of weekly task performance (actuals compared to plan). Inputs: Data of weekly Scheduled Performance Index (SPI). Outputs: XmR of weekly Scheduled Performance Index (SPI). Inputs: Data of weekly Cost Performance Index (CPI). Outputs: XmR of weekly Cost Performance Index (CPI). Inputs: Measures of each individual peer review on the peer review tool. Outputs: XmR of Peer Review Saves Per Size that alerts when the defect rate is high. Inputs: Measures of each individual peer review on the peer review tool. Outputs: XmR of Peer Review Prep Per Size that monitors the effort expended reviewing a work product prior to the peer review meeting. Inputs: Measures of each individual peer review on the peer review tool. Outputs: XmR of Peer Review Hours Per Size that monitors the total effort expended in a single peer review for a product.	Weekly task performance (actuals compared to plan) Weekly SPI. Weekly CPI. Peer review saves per size. Peer review preparation time per size. Peer review hours per size.	Not specified.	-	None.	-	Several programs have used the quantitative measures to overcome challenges and meet stakeholder objectives that previously would have been out of reach.	Yes.	Managing with quantitative measures has driven a two-fold increase in defects removed by peer reviews.

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2006.12	SPC, maintenance categories, expert heuristics, decision rules, Bayesian networks.	A model for building maintenance charts basing on initial effort estimations.	Inputs: Historical data on maintenance time efforts, two boolean variables indicating whether a given code fragment is part of a test class, or whether it was created as a result of source code restructuring. Outputs: Control charts of time efforts per week, in three different process phases.	Time efforts per week.	Maintenance.	-	Only the classification part: an experiment evaluating the performance of different models used for classifying efforts into maintenance.	The classification models were evaluated and compared to each other.	The maintenance charts and the warning mechanism presented are valuable solutions for software process assessment, helping the managers to easily interpret the evolution in time of maintenance efforts and to find the sources of scheduling problems.	No.	-
2006.16	SPC, Z-chart which assumes Poisson distribution, XmR chart.	Emphasis on processes rather than products.	Inputs: Historical data on test process. Outputs: Z chart bug rate within a department per release product. The value of each bug rate bi is normalized. Inputs: Historical data Peer Review Process. Outputs: XmR control chart of Review Speed per review meeting. Inputs: Historical data Peer Review Process. Outputs: Z control chart of Defect Density per review meeting. Inputs: Historical data Peer Review Process. Outputs: XmR control chart of Review Efficiency per review meeting group by projects.	Bug rate, early bug detection rate, review speed, defect density at peer review, early bug detection rate (for source code peer reviews, review efficiency.	Not specified.	-	None.	-	These examples show that SPC is applicable and useful in software development. SPC can provide navigation to which direction we should improve our processes in a measured way.	Yes.	Use not only product measurement but also process measurement. Put more emphasis on stability of the data rather than the amount of data. Consider and be aware of the psychological effect.
2006.24	SPC.	None.	Inputs: Number of defects, product size. Outputs: u-chart or XmR chart of defect density. Inputs: Rework effort, total effort. Outputs: XmR chart of Rework Percentage. Inputs: Number of defects found, inspection effort. Outputs: XmR chart of Inspection Performance.	Defect density. Rework effort. Review performance.	Emergent organizations.	-	A case study in an organization with data from seven projects with various characteristics.	-	SPC implementation is not a straightforward task in an emergent software organization. Relatively low-maturity processes may require some additional effort before the actual implementation. Nevertheless, these costs can be justified by the associated improvements in the processes.	No.	-

Table B9. Project process definition methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2014.03	Spiral concept, high maturity.	They proposed a model based on problems of the spiral model.	Inputs: Project goals and risks. Outputs: Project process with a lifecycle of two stages, seven most common risk patterns as examples.	–	Not specified.	–	4 examples of different risk patterns yielding different processes.	–	ISCM supports adapting and applying multiple processes as needed throughout a project, regardless of size, duration, or complexity.	No.	–

Table B10. Project estimation methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2007.14	Burn-down charts, burn-up charts, Order statistics, Bayesian statistic.	Using them to create a method to estimate how much additional time is needed to finalize planned work.	Inputs: Work planned in a burn-up chart format, work completed until the actual observation. Oututs: Sequence of estimates.	Work units planned, work units completed.	Not specified.	-	One example.	-	Preliminary experiments revealed that the approach performs well in practice. Project stakeholders were satisfied since they can realize the project progress at any time.	No. They do not detail its use.	-
2014.06	Earned Value Management (EVM), SPC.	The integration of historical cost performance data to EVM technique.	$CPIexp = ((EVacumproj + \sum_1^n(BACpn - EVacumpn) + \sum_1^n BACpn))\ /\ (ACacumproj + \sum_1^n(ACexppn - ACacumpn) + \sum_1^n ACexppn))$, where historical data from stable processes is considered.	Cost.	Not specified.	22 software development projects.	Average error.	With traditional EVM technique and real cost values.	The proposed technique was more accurate and precise than the traditional one.	No.	–
2012.10	Wideband Delphi, WBS, bottom-up, process database, process capability baseline.	They propose an estimation process.	Inputs: Project characteristics, process database, process capability baselines. Outputs: Activities to estimate effort including Wideband Delphi estimation, bottom-up estimation, consolidation, and re-estimation.	Effort.	Not specified.	–	Used in two projects.	With estimates generated before and with real effort.	New project estimates were less inaccurate than old project ones (4-7% vs 11-21%).	Yes, in two projects.	The inaccuracy of estimates dropped from 15% to 6%.
2012.22	SPC, moving range charts and run charts.	They used SPC to calculate training ROI considering process variation.	Inputs: Defects data. Outputs: Steps to calculate training ROI, including: determine historical sample, deploy and execute the process, evaluate data availability, estimate ROI, establish actual control limits, calculate ROI.	Defects.	Training interventions.	–	Case study at one company.	–	Applying SPC to ROI calculation provides better insight into the risks and benefits associated with a specific training.	Yes, by a CMMI level 3 software factory.	ROI for the training intervention was [21,007%, 2,935%] with an observed case of 690% for a six-month period.
2010.09	DMAIC, multiple linear regression, simulation.	None, merely their use.	Inputs: Statistical process model to predict productivity, performance baselines. Outputs: Project estimation with project goal, average, and upper and lower limits. The average comes from performance baselines, while others are calculated through simulations and the statistical model.	% defects in technical revisions, unit test coverage, defect density in systemic tests.	Not specified.	Data from the company.	They test the model on five projects of the organization to verify its efficacy in predicting final productivity.	With the percentage difference between planned productivity at the beginning and at the end of the project.	The model is a good way to obtain more precise estimates, to improve client satisfaction and to reduce variability of timeline, cost and productivity.	Yes, in 5 projects at a CMMI level 3 company.	Productivity estimation using the models is significantly more precise than traditional techniques such as, for example, the use of organization historical average.

Table B11. Root-cause analysis and problem resolution methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2011.06	CMMI, ISO 20000, 5 Why's, Pareto chart, Six Sigma, root cause analysis.	They present a corrective and preventive action method.	Inputs: Historical data. Outputs: Steps to correct and prevent problems including employing Pareto charts, gathering information and data, investigation, clarification and evaluation.	Defects.	Not specified.	One of the internal project of the company.	They created the process for an IT company.	–	It allows managers to track problems and prioritize by rescheduling problems. CAPA has a parallel control for both corrective and preventive actions.	No.	–
2011.10	PDCA, A3 problem-solving process.	The use of A3 problem-solving in 4 examples.	Inputs: Issues that impact many delivery teams to many customers. Outputs: A workshop involving people from all the affected teams to work through the A3 steps, called: plan, do, check, adjust.	Fix time of failed builds.	Not specified.	–	They applied the method to four examples in real projects.	–	The application of A3 problem-solving is a powerful tool for an individual project, which is amplified when used across projects with the involvement of senior management.	Yes, in four examples.	They believe successful projects will achieve all the objectives and troubled projects will fail on at least one of the objectives.

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2008. 14	Six Sigma, DMAIC.	They simplified the DMAIC model to address software development projects.	Inputs: Issues or problems. Outputs: Steps to address problems grouped into: define, measure, analyze, improve, and control phases.	–	Not specified.	–	They explained the method, but did not evaluate it.	–	MiniDMAIC can help organizations achieve higher maturity levels, increase customer satisfaction and reduce process variation.	No.	–

Table B12. Project management methods: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2004. 01	Six Sigma, SPC.	The steps proposed.	Inputs: A project needs. Outputs: A project completion after four steps, namely Identify, Define, Design and Optimize, and Verify.	Percentage of rework effort.	Not specified.	-	None.	-	The integrated approach uses statistical and other Six Sigma techniques that helps in meeting the customer demand for high quality and enables program predictability, and can result in program management excellence.	No.	-
2004. 02	CMM, QFD, Goal-driven Measurement Process, GQM, Simplified Quality Functional Deployment, ISO/IEC 9126, Simplified House of Quality.	Their use.	Inputs: A project needs. Outputs: Project measure analysis and diagnosis (SQFD) method, integrated with further steps to link QFD activities with the other related CMM activities: Identify customer's software quality needs, Quality characteristic and ranking, Determining the process voice and the product features, Evaluating the relationship values matrix between quality goals and process/product features, Definition of quality indicators: in Step 5, indicators are selected, Complement of project quality plan, Project's software products are measured and analyzed, and Analysis and diagnosis.	Five areas: size, effort, schedule, defect, and risk. Plus project staffing, duration of activities, and changes.	Software maintenance	-	Their use on four pilot projects.	They compared level 4 metrics with previous level 3 ones.	From the tailoringwork done on QFD and GDMP, they have learned that adapting a methodology to insert it in a methodological framework is a very delicate matter that must take into consideration the whole landscape and the foreseeable scenarios of use.	Yes.	A 5% increase in costs on each project can be expected owing to SPI activities.

Table B12. Project management methods and tools: Answers to research questions

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2017. 03	SPC, Fishbone analysis, correlation analysis, regression analysis, Monte Carlo Simulation and sensitivity analysis tools.	Their use on an organizational process asset library architecture to support high maturity.	Inputs: A new project. Outputs: An OPAL consisted of: Standard process library, Lifecycle model library, Tailoring guideline library, Measurement repository, Related document library.	Sales/ department revenue, labor utilization rate, delay rate, production efficiency, rate of requirement change, defects density in the acceptance.	Not specified.	-	This structure is implemented in an actual enterprise's OPAL and illustrated that it is beneficial for high maturity level process improvement.	-	This architecture serves the high maturity process improvements conveniently and it is with the extensibility and easily implemented by software enterprises.	No.	-
2010. 19	Project management, SPC, causal analysis.	The author put them together into steps to manage a project.	Inputs: A new project. Outputs: A step-by-step approach for monitoring projects and preventing defects, with steps for project initiation, recommended project organization structure, project execution and tracking, monitoring and control, forms / tools / checklist, defect prevention through defect pattern analysis, causal analysis, project closure.	Delivered defects, defects likely to be injected in each LC stage, productivity.	Not specified.	–	Examples and templates.	–	By employing quantitative project management techniques, organizations have not only improved their maturity in process deployment but also have achieved customer satisfaction objectives.	No.	–

ID	RQ3	RQ4	RQ5	RQ6	RQ7	RQ8	RQ9	RQ10	RQ11	RQ12	RQ13
2008.39	SPC, CMMI.	They describe models and a project management environment to support high-maturity project management.	Inputs: A new project. Outputs: Five basic process performance models and a project management environment to support high-maturity project management. The environment includes a log management system, a project management system, a quality monitor, and an enterprise process modeling system.	–	Not specified.	–	More than 27 organizations have adopted it.	–	More than 27 organizations in China have applied the method to improve their software capability maturity and CMMI appraisal.	Probably, but the paper does not detail this.	–
2007.29	TSP	Some adaptations to completely address CMMI requirements.	Inputs: Data already used on TSP. Outputs: Adaptations on TSP. They decided to track rework and the forecast completion date of its various work products. The team's EV tool computed the forecast completion date of the project and could also compute the forecast completion date of each of the project subparts. Rework time for this TSP team was defined as time recorded in the defect logs.	Rework, forecast completion date.	TSP.	Historical data from the company.	The GTACS team in 309th SMXG at Hill Air Force Base, used the TSP.	-	They adapted from and added to the TSP scripts, measures, and forms in ways that they believe can help other TSP teams also achieve this feat, as far as can be done by a single focus project.	Yes.	The team, successfully used the TSP in reaching their goal of CMMI Level 5.