Integration of Software Measurement Supporting Tools: A Mapping Study

Vinícius Soares Fonseca Monalessa Perini Barcellos Ricardo de Almeida Falbo

Ontology and Conceptual Modeling Research Group (NEMO)
Computer Science Department, Federal University of Espírito Santo
Vitória, ES, Brazil
{vsfonseca, monalessa, falbo}@inf.ufes.br

Abstract - **During software projects, it is necessary to collect, store and analyze data to support decision making at project and organizational levels. Software measurement is a key practice to process quality improvement and project management. Given the nature of measurement activities, supporting tools are essential. Different tools can be combined to support the measurement process and provide the necessary information for decision making. However, usually these tools are developed by different developers, at different points in time and without concern for integration. As a result, organizations have to deal with integration issues to allow tools communication and properly support the measurement process. This paper presents a study investigating in the literature initiatives involving tools integration to support software measurement. As a result, twelve proposals were analyzed and their characteristics are presented.**

Keywords - Software measurement; software measurement process; software measurement tools; integration; interoperability; systematic mapping

I. INTRODUCTION

Organizations use software measurement in several contexts. For example, in project management, measurement helps to develop realistic plans, monitor progress, identify problems and justify decisions [1]. In process improvement, measurement supports analyzing process behavior, identifying needs for improvement and predicting if processes will be able to achieve the established goals [2]. For this, data related to software processes, such as project management and testing processes, must be collected and analyzed.

Typically, organizations use different tools to support different processes, such as supporting tools for project management, requirements management, testing and bug tracking. Although, in general, these tools are not conceived aiming at supporting software measurement, many times they support collecting and storing useful data related to those processes (e.g., number of detected defects, time spent on activities, number of lines of code, etc.).

In order to properly support the software measurement process, providing consistent data to generate useful information, tools should be integrated. However, integration is a complex task. In general, each tool runs independently and implements its own data and behavioral models, which are not shared between different tools, leading to several conflicts [3].

(DOI reference number: 10.18293/SEKE2015-058)

Considering this scenario, we investigated the literature looking for initiatives involving tools integration to support software measurement. Aiming to reduce bias and ensure the study repeatability, the investigation was conducted as a systematic mapping. According to [4], a systematic mapping makes a broad study in a topic of a specific theme and aims to identify available evidence about that topic.

This paper is organized as follows: Section II briefly discusses software measurement and integration; Section III presents the research protocol used to guide the study; Section IV describes the obtained results; Section V discusses the results; and Section VI presents our final considerations.

II. BACKGROUND

A. Software Measurement

Software measurement (SM) is a primary support process for managing projects. It is also a key discipline in evaluating the quality of software products and the performance and capability of software processes. The software measurement process includes: measurement planning, measurement execution, and measurement evaluation [5].

For performing software measurement, initially, an organization must plan it. Based on its goals, the organization has to define which entities (processes, products and so on) are to be considered for software measurement and which of their properties (size, cost, time, etc.) are to be measured. The organization has also to define which measures are to be used to quantify those properties. For each measure, an operational definition should be specified, indicating, among others, how the measure must be collected and analyzed. Once planned, measurement can start. Measurement execution involves collecting data for the defined measures, storing and analyzing them. The data analysis provides information to decision making, supporting the identification of appropriate actions. Finally, the measurement process and its products should be evaluated in order to identify potential improvements [6].

B. Integration and Interoperability

Integration and interoperability are very related notions. *Integration* can be defined as the act of incorporating components into a complete set, conferring it some expected properties and creating synergy [3]. *Interoperability*, in turn, can be understood as the ability of applications or application components to exchange data and services [7]. Due to their interrelation, these terms are often used in an indistinct way [8]. In this paper, the term *integration* is adopted with a wider sense, covering both integration and interoperability meaning.

Izza [3] synthesizes integration approaches through four main dimensions: *scope*, which distinguishes between intra- and inter-enterprise integration; *viewpoint*, considering user, designer, and programmer views; *layer*, referring to data, service/message, and process integration; and *level*, which considers hardware, platform, syntactical, and semantic integration. For this paper, the two last dimensions are particularly relevant. Regarding integration layers, data integration deals with moving data between multiple data stores. Integration at this layer assumes bypassing the application logic and manipulating data directly in the database, through its native interface. Service/message integration addresses messages exchange between the integrated applications. Process integration views enterprises as a set of interrelated processes and it is responsible for handling message flows, implementing rules and defining the overall process execution. With respect to integration levels, syntactical integration encompasses the way data model and operation signatures are written down, while semantic integration encompasses the intended meaning of the concepts in a data schema or operation signature [3].

Challenges in applications integration arise, among others, from the fact that heterogeneous applications employ different data and behavioral models, leading to semantic conflicts. These conflicts occur whenever applications are built with different conceptualizations, which can impact the integration of data, services, and processes [8].

III. THE RESEARCH PROTOCOL

The study was performed following the approach defined in [4]. According to this approach, a systematic mapping involves: *planning*, when the research protocol is defined; *conducting*, when the protocol is executed and data are extracted, analyzed and recorded; and *reporting*, when the results are recorded and made available to potential interested parties. In this section we present the main parts of the research protocol used to perform the study.

Research Questions: the goal of this mapping study is to depict a general view of the current status of the research regarding tools integration to support software measurement. Table I presents the research questions that this mapping study aims to answer, as well as the rationale for considering them.

Search String: the search string was developed considering three groups of terms that were joined with the operator AND. The first group includes terms related to integration and interoperability. The second group includes terms related to software measurement. The third group includes terms related to tools and applications. Within the groups, we used the OR operator to allow synonyms. The following search string was used: ("integration" OR "integrated "OR "interoperability" OR "interoperable") AND ("software measurement" OR "software process measurement" OR "software project measurement" OR "software engineering measurement" OR "software product measurement") AND ("tool" OR "application" OR "system" OR "framework" OR "suite" OR "toolkit").

For establishing this search string, we performed some tests using different terms, logical connectors, and combinations among them. More restrictive strings excluded some important publications identified during the informal literature review that preceded the systematic mapping. These publications were used as control publications, meaning that the search string should be able to retrieve them. We decided to use a comprehensive string that provided better results in terms of number and relevance of the selected publications, even thought it had selected many publications that had to be eliminated in subsequent steps.

TABLE I. RESEARCH QUESTIONS

ID	Question	Rationale
RQ1	When and in which type of vehicle (journal / scientific event) have the publications been published?	Give an understanding on when and where the selected publications have been published.
RQ2	Which types of research have been done?	Identify the research type according to the classification defined by Wieringa et al. [9]: Evaluation Research; Proposal of Solution; Validation Research; Philosophical Paper; Opinion Paper; and Personal Experience Paper.
RQ3	Which types of tools have been integrated for supporting SM?	Identify the types of the integrated tools (e.g., project management tool, issue tracking tool, etc.) and verify whether a type is used in more than one proposal.
RQ4	Have the integrated tools been developed by the same group or organization?	Verify whether or not the initiatives have been integrating tools developed by the same group or organization. The purpose is to analyze if there is a trend in using tools developed by the same or by different groups.
RQ5	Which SM process activities (measurement planning, data collection, and data analysis) are supported by the integrated set of tools?	Identify which measurement activities are being supported by the initiatives, in order to evaluate the coverage of the resulting set of integrated tools. The activities considered are the two first activities established in [5] (measurement planning and measurement execution). Moreover, measurement execution was split for allowing us to verify if the tools support both data collection (which involves data collection itself and data storage) and data analysis, or only one of them.
RQ6	Which categories of measures are addressed by the proposal?	Identify which categories of measures (e.g., code measures, tests measures, etc.) have been considered, allowing us to analyze how specific or comprehensive is the measurement scope.
RQ7	In which layers (data, message/service or process) does the integration occur?	Identify the layers in which the integration is performed, considering the layers defined in [3]. The purpose is to analyze in which layer the integration initiatives have been focused on.
RQ8	In which level (syntactical or semantic) does the integration occur?	Identify the levels in which the integration is performed, considering the levels defined in [3]. The purpose is to analyze in which level the integration initiatives have been focused on.
RQ9	Does the proposal support measurement in the context of maturity models or standards? If so, which ones?	Identify which proposals support measurement in the context of maturity models (e.g., CMMI) or standards (e.g., ISO/IEC 9001), allowing us to verify whether supporting maturity models and standards has been a concern in SM tools integration initiatives.

Sources: the following six electronic databases were searched: IEEE Xplore (http://ieeexplore.ieee.org), ACM Digital Library (http://dl.acm.org), Springer Link (http://www.springerlink.com), Scopus (http://www.scopus.com), Science Direct (http://www.sciencedirect.com), and Engineering Village (http://www.engineeringvillage.com). They were selected based on other systematic reviews in the Software Engineering area.

Publication Selection: selection was performed in five steps*:*

<u>Step 1 (S1) - Preliminary selection and cataloging:</u> the search string was applied in the search mechanisms of the selected sources. Publication type was limited to papers from the Computer Science and Engineering area.

<u>Step 2 (S2) – Duplicates Removal</u>: studies indexed by more than one digital library were identified and the duplications were removed.

<u>Step 3 (S3) – Selection of Relevant Publications – First Filter</u>: selecting publications by applying a search string does not ensure that all selected publications are relevant, because such selection is restricted to syntactic aspects. Thus, the title, abstract and keywords of the selected publications were analyzed considering the following inclusion (IC) and exclusion (EC) selection criteria: (IC1) the publication presents information regarding integration among tools, applications or systems that support software measurement; (EC1) the publication does not have an abstract; (EC2) the publication is published as an abstract; and (EC3) the publication is not a primary study.

<u>Step 4 (S4) - Selection of Relevant Publications – Second Filter</u>: the full text of the publications selected in S3 was read with the purpose of identifying the ones that provide useful information. Thereby, the inclusion criterion IC1 was considered and also the following exclusion criteria: (EC4) the publication is not written in English; (EC5) the publication full text is not available; and (EC6) the publication is a copy or an older version of an already considered publication.

<u>Step 5 (S5) - Snowballing</u>: as suggested in [4], the references of publications selected in the study must be analyzed and, if some of them seems to present evidence related to the research topic, it should be assessed by the selection criteria and included in the study. Thus, in this step, references of the publications selected in S4 were investigated by applying the first and second filters.

IV. RESULTS

The systematic mapping considers publications until December 31st 2014. As a result of S1, 948 publications were obtained (357 from IEEE Xplore, 90 from Scopus, 257 from ACM, 8 from Science Direct, 49 from Engineering Village and 187 from Springer Link). After S2, 85 duplications were eliminated, achieving 863 publications. After S3, only 24 studies were selected (a reduction of approximately 97%). After S4, we achieved 8 studies. Applying snowballing (S5), 4 publications were added, reaching a total of 12 publications. Table II presents a brief description of each proposal. Following, we present the main results obtained for each research question.

<u>Publications source and year (RQ1)</u>: as Table II shows, the first study was published in 1988. Some studies were published since then, but research in the area has not been stable and presents two gaps (one between 1989 and 1996, and another between 2004 and 2009). Since 2010, it seems to be a more continuous research in the topic, with at least one work published per year. Most studies were published in scientific events (7) instead of journals (5).

<u>Research type (RQ2)</u>: all the analyzed studies include *proposals of solution.* Studies [11], [12], [13], [14], [15], [19], [20] and [21] are also categorized as *evaluation research*, since they have been applied into a production environment in at least one organization. Studies [10], [16] and [18] are also considered *validation research* due to the use of a prototype or experiment to evaluate the proposal.

<u>Integrated Tools (RQ3, RQ4)</u>: Table III presents the types of the tools being integrated in each proposal. Except [10], in which it was not possible to identify whether the tools were developed by the same group or not, all proposals integrated tools developed by different groups.

<u>Measurement activities and Measures (RQ5, RQ6)</u>: only four proposals ([10], [12], [13] and [20]) address *Measurement Planning*. *Data Collection* and *Data Analysis*, in turn, are addressed by all proposals. Table III presents the categories of the measures addressed by the proposals.

<u>Integration layers (RQ7)</u>: seven proposals ([12], [13], [14], [15], [18], [20] and [21]) address integration only in *data layer*; four proposals ([10], [16], [17], and [19]) address integration only in the *message layer*; and one proposal ([11]) addresses integration in both *data* and *message layers.* None of the proposals address the *process layer.*

<u>Semantic aspects (RQ8)</u>: only [16] addresses integration in the semantic level; the others address integration in the syntactical level. In [16], Ghezzi and Gall use ontologies implemented in OWL to define and represent the data consumed and produced by the services of the integrated tools.

<u>Maturity Models/Standards (RQ9)</u>: only two studies mention maturity models/standards and both of them refer to CMMI [22]. [15] uses CMMI measurement practices to define the measurement program supported by the proposed framework. [20] was conducted at a CMMI Level 3 organization.

TABLE II. PROPOSALS INVOLVING TOOLS INTEGRATION FOR SUPPORTING SOFTWARE MEASUREMENT

Proposal	Year/Vehicle	Description
TAME [10]	1988 Journal	TAME (Tailoring A Measurement Environment) system is an Integrated Software Engineering Environment that is composed by several components. TAME integrates three measurement tools that capture data from Ada source and generate measures.
Tool support for SM [11]	1997 Journal	This approach uses a set of integrated tools in order to support software measurement and quality improvement. A tool that supports tree-modeling analysis (S-PLUS) is the central analysis tool. Other tools are used for data gathering, analysis, and result presentation. The tools are connected to S-PLUS, either as an information consumer or as information providers. The integration is achieved through the adoption of external rules for data contents and formats (to ensure tools interoperability), the usage of common tools for multiple purposes, and the usage of utility programs that convert data for tools interoperability.
GQM tool [12]	2000 Journal	GQM tool supports measures definition, data collection, analysis, and feedback. It has interface with a configuration management system and other measurement tools. The integration with the configuration management system is performed through a data link between their relational databases. The integration with the other measurement tools is done by developing an XML translator for each tool, allowing the translation of the native data format to XML format.
MetriFlame [13]	2001 Symposium	MetriFlame is a measurement automation tool based on GQM that uses existing data recorded during software development process. It has components for collecting and converting measurement data from various tools, spreadsheets and databases. In the paper, the components are not detailed and no further information about integration is given.
A Decision Support System [14]	2003 Workshop	The Decision Support System was developed at IBM for tracking and using software measures aiming to enable executives to make better informed decisions in supporting their products. It captures (from different host systems) data regarding customer support, critical situations and customer satisfaction and integrates these data into a data warehouse.
SM in a CI Environment [15]	2010 Conference	It uses a Continuous Integration (CI)[1] engine in order to automate measurement data extraction. It follows CMMI Measurement and Analysis process area practices and GQ(I)M concepts for selecting relevant measures. Data collection is done by several tools. After extraction, data are consolidated in a XML file that is stored into a relational database until an ETL (Extract, Transform, and Load) process run and load data into a data warehouse. An OLAP tool is used to analyze data.
SOFAS [16]	2011 Conference	SOFAS is a platform that offers software analysis services to allow for interoperability among analysis tools. It is made up of three main constituents: Software Analysis Web Services, which provides a catalogue of services for data analysis; a Software Analysis Broker, acting as the services manager and the interface between the services and the users; and Software Analysis Ontologies, which defines and represents the data consumed and produced by the different services.
Dione [17]	2012 Symposium	Dione is a Java web application whose majors functions are: i) build a measurement repository that contains product and process measures as well as information about defected software components; ii) analyze trends in measures and issues using chart and report configurations; and iii) construct and calibrate customized defect prediction models to predict defect proneness of a software product version or release. It collects data from several tools and uses a smart client to connect with software development artifacts and automatically extract measures. It also supports integration with other tools through web services.
QualitySpy [18]	2012 Journal	QualitySpy is a framework for monitoring the software development process. It collects raw data from several integrated tools as well as from the source code. The collected data and reports are available in a reporting module implemented as light web client, which communicates with the server using Representational State Transfer (REST).
The 3C Approach [19]	2012 Workshop	The 3C Approach is an extension to the CI practice and addresses Continuous Measurement and Continuous Improvement as subsequent activities to Continuous Integration. Several Java tools and a version control system were integrated into the CI engine CruiseControl, allowing collection of measures related to source code and test coverage.
ASSIST [20]	2013 Conference	ASSIST is an integrated tool developed by a CMMI level 3 organization. It adopts GQ(I)M approach and is connected with commercial software suites for project management, issue tracking and enterprise resource planning. ASSIST uses a low-level integration strategy based on SQL because all the tools involved depend on a relational database management system. It allows automatic data collection from the integrated tools, data import from spreadsheets and manual data entry.
DePress [21]	2014 Journal	DePress is an open source, extensible framework for software measurement and data integration which can be used for prediction purposes (e.g., defect prediction, effort prediction) and software changes analysis (e.g., release notes, bug statistics). It supports the integrated use (through KNIME Framework) of the issue tracking systems JIRA and Bugzilla, the software configuration management systems SVN and GIT, and the measurement tools Judy, JaCoCo, EclipseMetrics, CheckStyle and PMD.

TABLE III. TYPES OF INTEGRATED TOOLS AND CATEGORIES OF MEASURES ADDRESSED BY THE INTEGRATION INITIATIVES.

Pub.	Types of the tools being integrated	Measure categories
[10]	Code Measurement, Tests	code, size, test
[11]	Code Measurement, Tests, Configuration Management, Issue Tracking, Modeling, Presentation/Reporting, Reverse Engineering	code, size, test, defects, changes, design, transactions
[12]	Code Measurement, Configuration Management	code, size, defects
[13]	Configuration Management, Project Management, Document Management, Databases, Spreadsheets	it depends on the data available in the integrated tools, databases and spreadsheets
[14]	Customer Management, OLAP Tool	problem, product quality, customer satisfaction
[15]	Code Measurement, Tests, Continuous Integration, Build Automation, OLAP Tool	code, size, test
[16]	Code Measurement, Configuration Management, Issue Tracking	code, size
[17]	Code Measurement, Configuration Management, Issue Tracking, Presentation/Reporting	code, size, defects
[18]	Code Measurement, Configuration Management, Issue Tracking, Continuous Integration	code
[19]	Code Measurement, Tests, Configuration Management, Continuous Integration, Build Automation, Presentation/Reporting	code, size, test
[20]	Issue tracking, Project management, Enterprise Resource Planning, Spreadsheets	code, size and other measures depending on the data available in the integrated tools and spreadsheets
[21]	Code Measurement, Configuration Management, Issue Tracking, Presentation/Reporting, Defect Prediction, Data Mining Tool, Security, Statistics, Spreadsheets	code, defects, time, issue

[1]CI is a practice for continuous integration of new source code into the base code, including automated compile, build and running of tests [19].

V. Discussion

This section provides some additional discussion about the results presented in the previous section.

Concerning the measurement activities, we noticed that all proposals that support *Measurement Planning* ([10], [12], [13] and [20]) use GQM (Goal-Question-Metric) paradigm [23] or one of its variations. GQM is based on the idea that measurement should be goal-oriented, i.e., data must be collected based on an explicitly documented rationale [12]. Thus, GQM addresses measurement planning by guiding measures definition from established goals. Since GQM has been successfully adopted in software measurement initiatives for years, its usage by the proposals that address measurement planning was expected.

All proposals support *Data Collection* and *Data Analysis*. Regarding data collection, data are automatically captured by the tools or input by using their interface. As for data analysis, [11] supports the use of collected data to analyze software reliability; [10], [12], [13], [17] and [19] allow to analyze whether the established goals have been achieved; [14] supports the analysis of customer satisfaction; and [17] and [21] support analysis aiming at defect prediction.

With respect to the integrated tools, although we did not list in this paper the tools involved in each proposal, they are diverse. There are proposals integrating commercial tools (e.g. [11], [13], [20]), open source tools (e.g. [15], [16], [18], [19], [21]) and in-house developed tools (e.g. [10], [12], [20]). We also noticed that some proposals focus specifically on integrating existing tools (e.g. [11], [15], [16], [18], [19]), while others address the development of a whole integrated tool (e.g. [10], [20]). Moreover, we noticed that in some initiatives ([11], [12], [13], [15], [16], [20], [21]) measurement process support was the main motivation for integrating tools, while in others ([10], [14], [17], [18], [19]) the measurement support was achieved as a consequence of the tools integration. For instance, in [18] tools are integrated to support software development process monitoring and, as a consequence of the integration, software measurement was also supported. The variety of tools that can be used to support measurement increases the relevance of integration in this domain, because organizations could choose the tools that are more suitable for their needs and work on their integration. .

Even though the integrated tools are diverse, it is possible to notice a predominance of code-related tools. Code Measurement, Issue Tracking, and Configuration Management tools are integrated in several proposals (9, 7 and 6 proposals, respectively). It might be a consequence of these types of tools being prone to automatic collection of measures. Besides, some of them depend on others to provide information. For instance, since source code is usually stored in a Configuration Management system, the presence of a Code Measurement tool usually implies the presence of a Configuration Management tool.

Considering that code-related tools were integrated in most proposals, it was expected that code measures (e.g., cyclomatic complexity, number of methods) would be addressed by most proposals. 10 of the 12 studies address them. Taking the types of integrated tools and measures into account, except [13] and [20], which have a more comprehensive scope, the integration initiatives usually address a specific measurement scope (e.g. coding, customer support).

Analyzing the integration layers addressed, we noticed that the proposals deal with *data* and *message layer*, while *process layer* is not addressed. We believe this is due the fact that *process layer* integration (commonly referred to as Business Process Integration) constitutes the most complex integration approach [3]. It views an enterprise/organization as a set of interrelated business processes and not merely islands of information, dealing with message flows, rules and process execution. *Message layer* is addressed, but only by few proposals. Message layer integration requires tool communication by means of message exchange between the tools. If the integrated tools are not able to communicate by means of messages, integration in this layer demands extra effort, especially if tools were not developed by the group performing the integration (this is the case in most proposals). In this sense, according to [20], a low level of integration is preferred when integrating with commercial tools, because it does not involve any code development or modification at the commercial tools' side. All studies that integrate commercial tools ([11], [12], [13], [20]) are limited to data integration.

As for semantic integration, only [16] considers semantic aspects in the integration. Neglecting semantics during an integration initiative is a serious issue, since many semantic problems can occur, such as the ones called “false agreement”, which are described in [28] and include: the use of equivalent terms with different meaning; the use of equivalent terms with partially equivalent meaning; the use of different terms with equivalent meaning; and the use of different terms with a certain degree of equivalence. For addressing these problems, ontologies can be used to establish a common conceptualization about the domain in order to support communication and tools integration [24].

Since none of the proposals presented the method followed to perform the integration, we concluded that they have used ad-hoc approaches for integrating the tools. Not using a systematic approach for performing the integration, despite the existence of systematic approaches in the literature (e.g. [24], [25], [26]), can be seen as a gap regarding methodological aspects. Systematic approaches can structure the integration process into different levels of abstractions and define guidelines on how to perform the various integration activities. This is essential for establishing an engineering approach for application integration [27].

VI. Final Considerations

This paper presented the main results of a systematic mapping about initiatives involving tools integration for supporting software measurement. A total of 952 publications were analyzed and 12 proposals involving tools integration for supporting software measurement were found.

According to [29], a mapping study gives an idea of shortcomings in existing evidence, which becomes a basis for future studies. Therefore, the main objective of this mapping was to analyze the proposals and provide a general view of the current status of the research regarding tools integration for supporting software measurement. Summarizing, the analyzed proposals address measurement execution (data collection and analysis), but most of them do not address measurement planning. Integration in the data layer is most common, although some proposals deal with integration in the message layer. They predominantly integrate coding-related tools and address code measures. Supporting maturity models/standards have not been a concern. Finally, only one proposal considers semantic aspects and, apparently, none of the proposals used a systematic approach to perform integration.

This panorama reveals some gaps in the research regarding tools integration for supporting software measurement. We can highlight the following: (i) lack of concern with semantics; (ii) limited coverage with respect to the measurement process or the measure categories addressed by the integrated tool suite; (iii) lack of alignment to quality-related standards and maturity models; (iv) failure to consider a holist view of the (software) process, leading to the absence of integration in the process layer. Taking these gaps into account, we are now working on an integration of measurement supporting tools following a systematic approach aiming at overcoming these gaps.

ACKNOWLEDGMENT

This research is funded by the Brazilian Research Funding Agency CNPq (Processes 485368/2013-7 and 461777/2014-2).

REFERENCES

[1] J. McGarry, D. Card, C. Jones, B. Layman, E. Clark, J. Dean, and F. Hall, “Practical Software Measurement: Objective information for decision makers," Addison Wesley, Boston, USA, 2002.

[2] W. A. Florac, A. D. Carleton, “Measuring the software process: statistical process control for software process improvement,"Addison Wesley, Boston, USA, 1999.

[3] S. Izza, “Integration of industrial information systems: from syntactic to semantic integration approaches,” Enterp. Inf. Syst., vol. 3, no. 1, pp. 1–57, Feb. 2009.

[4] B. Kitchenham and S. Charters, “Guidelines for performing systematic literature reviews in software engineering," TR EBSE-2007-01, School of Computer Science and Mathematics, Keele University, 2007.

[5] ISO/IEC 15939, “Systems and Software Engineering – Measurement Process”, 2007.

[6] M. Barcellos, R. A. Falbo, and A. R. Rocha, “Establishing a well-founded conceptualization about software measurement in high maturity levels,” in Proc. of the 7th International Conference on the Quality of Information and Communications Technology, 2010, pp. 467–472.

[7] P. Wegner, “Interoperability,” in ACM Computing Survey, 28 (1), 1996, pp. 285–287.

[8] J. C. Nardi, R. A. Falbo, and J. P. A. Almeida, “A panorama of the semantic EAI initiatives and the adoption of ontologies by these initiatives,” in Proc. of the IWEI 2013, LNBIP 144, Lecture Notes in Business Information Processing, 2013, pp. 198–211.

[9] R. Wieringa, N. Maiden, N. Mead, and C. Rolland, “Requirements engineering paper classification and evaluation criteria: a proposal and a discussion,” Requir. Eng., vol. 11, no. 1, pp. 102–107, Nov. 2005.

[10] V. R. Basili and H. D. Rombach, “The TAME project: towards improvement-oriented software environments,” *IEEE Trans. Softw. Eng.*, vol. 14, no. 6, pp. 758–773, Jun. 1988.

[11] J. Tian, J. Troster, and J. Palma, “Tool support for software measurement, analysis and improvement,” J. Syst. Softw., vol. 39, no. 2, pp. 165–178, Nov. 1997.

[12] L. Lavazza, “Providing automated support for the GQM measurement process,” IEEE Softw., vol. 17, no. 3, pp. 56–62, 2000.

[13] S. Komi-Sirvio, P. Parviainen, and J. Ronkainen, “Measurement automation: methodological background and practical solutions a multiple case study,” in Proc. of the 7th International Software Metrics Symposium, 2001, pp. 306–316.

[14] S. Chulani, B. Ray, P. Santhanam, and R. Leszkowicz, “Metrics for managing customer view of software quality,” in Proc. of the 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (IEEE Cat. No.03EX717), 2003, pp. 189–198.

[15] G. de S. P. Moreira, R. P. Mellado, D. A. Montini, L. A. V. Dias, and A. M. da Cunha, “Software product measurement and analysis in a continuous integration environment,” in Proc. of the 7th International Conference on Information Technology: New Generations, 2010, pp. 1177–1182.

[16] G. Ghezzi, H. C. Gall, “SOFAS: A Lightweight Architecture for Software Analysis as a Service,” in Proc. of the Ninth Working IEEE/IFIP Conference on Software Architecture, 2011, pp. 93–102.

[17] B. Caglayan, A. T. Misirli, G. Calikli, A. Bener, T. Aytac, and B. Turhan, “Dione: an integrated measurement and defect prediction solution,” in Proc. of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering - FSE ’12, 2012, pp. 1–4.

[18] M. Jureczko, J. Magott, “QualitySpy: a framework for monitoring software development processes,” in Journal of Theoretical and Applied Computer Science, v. 6, n. 1, 2012, pp. 35–45.

[19] A. Janus, R. Dumke, A. Schmietendorf, and J. Jager, “The 3C approach for agile quality assurance,” in Proc. of the 3rd International Workshop on Emerging Trends in Software Metrics (WETSoM), 2012, pp. 9–13.

[20] B. Keser, T. Iyidogan, and B. Ozkan, “ASSIST: an integrated measurement tool,” in Proc. of the Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement, 2013, pp. 237–242.

[21] L. Madeyski, and M. Majchrzak,“Software measurement and defect prediction with DePress extensible framework”. Foundations of Computing and Decision Sciences, v. 39, n. 4, 2014, p. 249–270.

[22] CMMI-DEV, “Improving processes for better products and services,” in CMMI for Development, Version 1.3, CMU/SEI-2010-TR33, SEI, Carnegie Mellon University, Pittsburgh, 2010.

[23] V. R. Basili, H. D. Rombach, and G. Caldiera, “Goal Question Metric paradigm”, Encyclopedia of Software Engineering, 2 Volume Set, John Wiley & Sons, Inc., 2004.

[24] R. F. Calhau and R. A. Falbo, “An ontology-based approach for semantic integration,” in Proc. of the 14th IEEE International Enterprise Distributed Object Computing Conference, 2010, pp. 111–120.

[25] C. Liu, J. Wang, Y. Wen, and Y. Han, “A unified data and service integration approach for dynamic business collaboration,” in IEEE 1st International Conference on Services Economics, 2012, pp. 54–61.

[26] W. J. Yan, P. S. Tan, and E. W. Lee, “A web services-enabled B2B integration approach for SMEs,” in Proc. of the 6th IEEE International Conference on Industrial Informatics, 2008, no. Indin, pp. 774–779.

[27] J. C. Nardi, R. A. Falbo, and J. P. A. Almeida, “Foundational ontologies for semantic integration in EAI: a systematic literature review,” in I3E 2013, IFIP Advances in Information and Communication Technology, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 238-249.

[28] S. V.Pokraev, “Model-driven Semantic Integration of Service-Oriented Applications,” PhD thesis, University of Twente, 2009.

[29] B. A. Kitchenham, D. Budgen, and O. P. Brereton, “Using mapping studies as the basis for further research: a participant-observer case study,” Journal of Information and Software Technology, 53, 2011, pp. 638-651.