An Ontology-Based Approach for Software Measurement Systems Integration

Vinícius Soares Fonseca, Monalessa Perini Barcellos, Ricardo de Almeida Falbo

Ontology and Conceptual Modeling Research Group (NEMO), Department of Computer Science, Federal University of Espírito Santo – Vitória – ES – Brazil

{vsfonseca, monalessa, falbo}@inf.ufes.br

Abstract. *Software measurement is performed in the context of various software processes that, generally, have different supporting tools. Thus, organizations have to deal with integration issues to enable the communication between tools and to properly support the measurement process. A key factor for integration is that tools share a common understanding regarding the meaning of the exchanged terms and services, i.e., it is important to deal with integration at the semantic level. Ontologies have been acknowledged as an important means to address semantic integration. In this paper we present the Ontology-Based Approach for Measurement Systems Integration (OBA-MSI), an approach that uses ontologies as a basis to integrate tools aiming at supporting the software measurement process. In order to evaluate OBA-MSI, it was applied to integrate tools in a software development organization.*

1. Introduction

Software measurement is a process applied by organizations in several contexts. For instance, in project management, measurement is used to help to develop realistic plans, monitor project progress, identify problems and justify decisions [McGarry *et al*. 2002]. In process improvement initiatives, measurement supports analyzing process behavior, identifying needs for improvement and predicting if processes will be able to achieve the established goals [Florac and Carleton 1997].

Typically, organizations use different tools to support different processes. For example, schedule and budget tools are used to support project management, and development environments and version control systems are used to support coding and source code management. Although these tools are not usually conceived to support software measurement, many times they store useful data related to the supported processes (e.g., number of defects, time and cost spent on activities, etc.). In order to properly support the software measurement process, tools must be integrated, but this is not an easy task. The heterogeneity between systems is the major difficulty. 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 [Izza 2009].

Semantic conflicts occur when applications use different meanings to the same information item, i.e., when information items seem to have the same meaning, but they do not. To reduce these conflicts, integration initiatives should address semantic issues. Ontologies can be used as an interlingua to map the concepts used by different applications, enabling data and services understanding [Calhau and Falbo 2010].

Considering that, the purpose of the work described in this paper is to define an approach for integrating measurement tools aiming to support the software measurement process. We started by investigating the state-of-art through a systematic mapping and a systematic literature review [Fonseca *et al*. 2015a, 2015b]. 12 initiatives were identified. By analyzing them, we noticed a lack of concern with semantics and failure to consider a holist view of the software measurement process. Besides, none of the found initiatives followed a systematic approach to integrate tools. Since integration is not a trivial task, it is important to adopt an approach that helps deal with the complexity of the task by providing well-established steps, separation of concerns and reduction of subjectivity. Taking these gaps into account, we developed the *Ontology-Based Approach for Measurement Systems Integration* (OBA-MSI), a systematic approach that uses the Reference Software Measurement Ontology (RSMO) [Barcellos *et al*. 2013] and the Software Measurement Task Ontology (SMTO) [Barcellos and Falbo 2013] to guide applications integration to support software measurement process. OBA-MSI extends the *Ontology-Based Approach for Semantic Integration* (OBA-SI) [Calhau and Falbo 2010], which can be applied to carry semantic integration of applications in any domain. However, applications integration in the software measurement domain has some peculiarities that are not properly addressed by OBA-SI. Finally, aiming to evaluate OBA-MSI, we applied it to integrate tools to support the measurement process in a software development organization and ran a survey to obtain feedback from the organization development team and manager.

This paper is organized as follows: Section 2 addresses aspects related to software measurement and integration, and introduces OBA-SI, the approach from which OBA-MSI was developed; Section 3 presents OBA-MSI; Section 4 discusses the use of OBA-MSI to integrate tools in a software development organization; Section 5 covers related works; and Section 6 presents our final considerations.

2. Background

2.1 Software Measurement

Software measurement is the continuous process of defining, collecting, and analyzing data regarding software processes and products in order to understand and control them, as well as supply meaningful information to their improvement [Solingen and Berghout 1999]. It is a primary support process for managing projects, and it is also a key discipline in evaluating the quality of software products and the performance and capability of organizational software processes [ISO/IEC 2007]. The software measurement process includes: measurement planning, measurement execution, and measurement evaluation [ISO/IEC 2007].

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. Data analysis provides information to decision making, supporting the identification of appropriate actions. Finally, the measurement process and its products should be evaluated to identify potential improvements [Barcellos *et al.* 2013].

2.2 Integration and Interoperability

Integration can be defined as the act of incorporating components into a complete set, conferring it some expected properties. The components are combined in a way to form a new system constituting a whole and creating synergy [Izza 2009]. *Interoperability*, in turn, can be understood as the ability of applications or application components to exchange data and services [Wegner 1996]. It provides two or more business entities with the ability of exchanging or sharing information (wherever it is and at any time) and of using functionality of one another in a distributed and heterogeneous environment. It preserves component systems as they are. Due to the interrelation between the terms integration and interoperability, they are often used in an indistinct way [Nardi *et al.* 2013a]. In this paper, the term integration is adopted in a broader sense, covering both integration and interoperability meaning.

Integration can be performed considering different dimensions. Izza (2009) proposed a framework synthesizing integration approaches through four main dimensions: scope, viewpoint, layer and level. *Scope* dimension distinguishes two main approaches: *intra-enterprise* and *inter-enterprise* integration. The first one concerns scenarios that imply internal enterprise applications. The second one aims to connect applications from different partners. *Viewpoint* dimension, in turn, considers three viewpoints: *user's view* (external), concerning the different views from domain experts and business users; *designer's view* (conceptual), referring to the different models used during information system design; and *programmer's view* (internal), which regards information system implementation.

As for *layers*, integration can address one or several information system layers. *Data integration* deals with moving or federating 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. *Message or service integration* addresses messages exchange between the integrated applications. Any tier of an application, such as GUI, application logic or database, can originate or consume the message. *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. It constitutes the most complex integration approach.

Regarding integration *levels*, four main levels can be distinguished: *hardware*, *platform*, *syntactical* and *semantic* levels. Hardware level covers differences in computer hardware, networks, etc. Platform level encompasses differences in operating system, database platform, etc. Syntactical level addresses the way the data model and operation signatures are written down. Semantic level deals with the intended meaning of the concepts in a data schema or operation signature. Each level depends on the previous one, so it is not possible to consider semantics if syntax is not considered yet.

2.3 Ontology-Based Approach for Semantic Integration (OBA-SI)

OBA-SI [Calhau and Falbo 2010] is an approach that uses ontologies to address semantic integration problems in the context of tool integration. It considers the integration process as a software development process and, as such, it is composed by requirement elicitation, analysis, design, implementation, tests and deployment phases. OBA-SI is centered on the analysis phase, wherein semantics must be defined. It is necessary to establish semantic agreement before designing and coding any integration solution, i.e., integration at conceptual level should not depend on any technology and specific solution. To achieve this, the structural and behavioral conceptual models of the applications are compared with the aid of domain and task ontologies, used for assigning semantics to the items shared between systems. Figure 1 shows OBA-SI integration process.

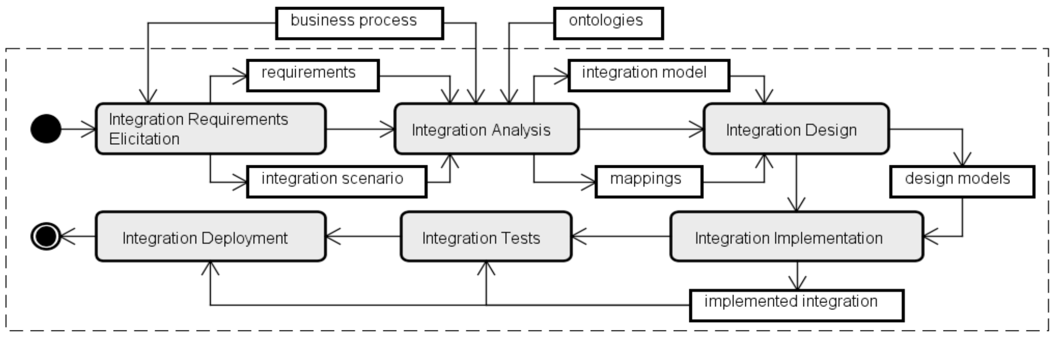


Figure 1. OBA-SI process [Calhau and Falbo 2010].

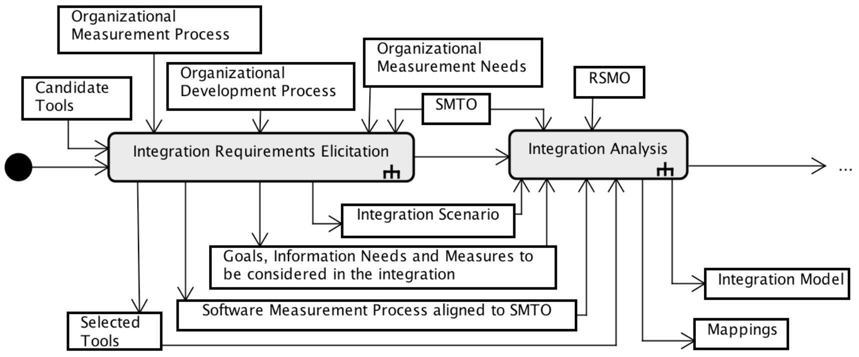
The integration starts with the *Integration Requirements Elicitation* phase, when the integration requirements and goals must be established. In this phase the integration scenario is produced, indicating the business process activities that will be supported by the integration initiative, the tools that will be integrated to support those activities, the domains involved into the integration scenario and the generic tasks related to the integration. Next, in the *Integration Analysis* phase, integration requirements are analyzed and modeled, features to be provided and concepts involved are specified, and the overall behavior of the integrated set of systems is defined. The output of this phase is the integration model, whose purpose is to model structural and behavioral aspects of the integration at conceptual level, taking the specified requirements into account. This phase starts by getting the structural and behavioral models of the selected tools. Then, domain and task reference ontologies are selected (or developed) and vertical mappings (VMs) between tools' elements (e.g., concepts and relations) and ontologies' elements are established aiming to assign meaning to the tools' elements by relating them to elements in the reference ontologies. Once VMs are established, the integration model is built based on the ontologies and the tools' models in a way that each element of the integration model has a meaning. Next, horizontal mappings (HMs) between the elements of the tools and of the integration model are established. HMs focus on the integration scenario, defining how the tools will be seen in the integration solution and how the interaction between them will occur. Both structural and behavioral mappings should be done. Structural mappings concern the data integration layer, while behavioral mappings regard process and service integration layers.

Last, there are the design, implementation, tests and deployment phases. There are several ways of building an integration solution, thus OBA-SI does not commit to any specific solution but proposes some guidelines to assure that semantics established in the integration analysis phase is maintained during the next phases. The tools can be integrated without being changed. In this context, typically a mediator is used for interconnecting tools and it has an overview of the integrated systems. The integration model is critical for the mediator because it provides information to design the mediator and the communication between it and the integrated tools.

3. Ontology-Based Approach for Measurement Systems Integration (OBA-MSI)

OBA-MSI extends OBA-SI and, as such, uses reference ontologies as basis to perform semantic integration. In OBA-MSI semantic integration is carried out based on the Reference Software Measurement Ontology (RSMO) [Barcellos *et al.* 2013], which describes a conceptualization to the software measurement domain, and on the Software Measurement Task Ontology (SMTO) [Barcellos and Falbo 2013], which describes the main activities of the software measurement process, their inputs and outputs, being consistent with RSMO. The main extension occurs in the requirements elicitation phase, which is detailed and conducted by using a goal-based approach that follows GQM [Basili *et al.* 1994] principles to guide software measurement aligned to organizational goals. Besides, the extension helps organizations to define an appropriate measurement process (if the organization does not have a measurement process) or to improve it (if the organization has a measurement processes).

The integration process defined by OBA-MSI includes the same phases defined by OBA-SI with differences in *Integration Requirements Elicitation* and *Integration Analysis* phases. Figure 2 presents these two phases in OBA-MSI. After the figure, these phases are described. The other phases are not shown because they are the same as the ones defined in OBA-SI.

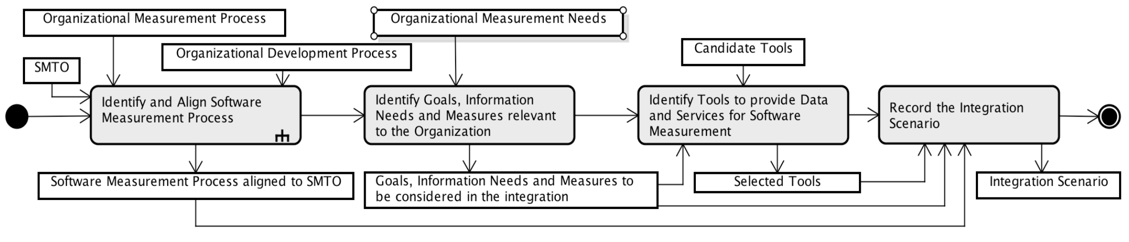


Organizational Measurement Process
Organizational Measurement Needs
Candidate Tools
Organizational Development Process
RSMO
SMTO
Integration Requirements Elicitation
Integration Analysis
...
Integration Scenario
Goals, Information Needs and Measures to be considered in the integration
Integration Model
Software Measurement Process aligned to SMTO
Selected Tools
Mappings

Figure 2. First phases of OBA-MSI process.

3.1 Integration Requirements Elicitation

In the ***Integration Requirements Elicitation*** phase, OBA-MSI uses a goal-based approach to assure that measurement is aligned to organizational needs. By doing this, the organization will measure only what really matters and will be prevented from collecting useless data not able to provide information for decision making. Besides, OBA-MSI advocates that the software measurement process should be appropriately defined in order to be supported by the integrated solution. Thus, the organizational measurement process should be aligned to the measurement process established in SMTO. Figure 3 details the *Integration Requirements Elicitation* phase.



Organizational Measurement Process
Organizational Measurement Needs
SMTO
Organizational Development Process
Candidate Tools
Identify and Align Software Measurement Process
Identify Goals, Information Needs and Measures relevant to the Organization
Identify Tools to provide Data and Services for Software Measurement
Record the Integration Scenario
Software Measurement Process aligned to SMTO
Goals, Information Needs and Measures to be considered in the integration
Selected Tools
Integration Scenario

Figure 3. OBA-MSI – Integration Requirements Elicitation phase.

Identify and Align Software Measurement Process activity deals with the alignment of the organizational software measurement process to SMTO to ensure that the organizational software measurement process includes all activities necessary for measurement to be carried out properly and that these activities are properly defined. Contrasting with OBA-SI, OBA-MSI brings the alignment between the measurement process and the task ontology to the beginning of the semantic integration effort, intending not just an integrated set of tools to support the organizational measurement process, but also the process improvement. It turns the integration initiative (and the resulting integrated set of tools) an agent of change for the organization, allowing the measurement process improvement. This activity is composed of four sub-activities (not shown in Figure 3) that are described in the following and highlighted in *underlined italic* font.

For aligning the organizational software measurement process to SMTO, it is necessary to *Verify the Existence of Organizational Software Measurement Process*. Some organizations have a defined measurement process, while others carry out measurement activities implicitly during their development process. There are also organizations that do not perform software measurement and will start the practice from the tool integration initiative. An organization with a defined measurement process should *Align the Software Measurement Process to SMTO*. If the organization does not have a defined measurement process, but performs measurement along its development process, then it should *Identify Measurement Activities in the Organizational Software Development Process*, (i.e., activities in which measurement takes place must be identified) and, then, the measurement process must be defined and aligned to SMTO. Finally, if the organization does not have a defined measurement process and does not perform software measurement, it should *Define the Software Measurement Process from SMTO*, i.e., SMTO must be used as a basis to establish the organizational software measurement process.

Identify Measurement Activities in the Organizational Software Development Process requires the organizational development process to be carefully analyzed in order to identify activities wherein there is some measurement-related activity. For instance, if the development process has an activity "Code and Test Software" in which data regarding source code is collected, measurements occur in the context of that activity. Some techniques can be applied, such as interviews and meetings, in a way that information obtained about the development process allows identifying activities related to measurement. Ideally, to perform this activity, the software development process should be formally defined (e.g., textually or in a diagram). If the organization does not have a defined development process, it should be defined in order to allow for identification of measurement activities. Once the activities related to measurement are identified, they should be analyzed to verify if they can be decomposed to turn measurement activities explicit. For instance, the "Code and Test Software" activity cited before could be split into "Code and Test Software" and "Collect Source Code Data", being the last one devoted to perform measurements regarding source code.

The identified measurement activities (if the organization performs measurement in the context of the development process activities) or the activities of the organizational software measurement process (if the organization has a defined measurement process) must be aligned to SMTO. To *Align the Software Measurement Process to SMTO,* the measurement activities must be mapped to activities of the measurement process established in SMTO. After that, the measurement process must be defined by following the three main SMTO activities (Plan Measurement, Perform Measurements and Analyze Measurements). Each activity of the measurement process must be detailed based on the organizational measurement activities or SMTO activities (when measurement activities are missing in the organizational process).

Once the software measurement process is consistent with SMTO, it is necessary to ***Identify Goals, Information Needs and Measures relevant to the Organization***. This activity is responsible for ensuring the alignment between measurement and organization's goals, and takes into account the principle that software measurement must be goal-based for providing useful information. In this sense, in order to integrate tools and aggregate value to the organization, the goals should be established before starting measure. For this, organization's business goals relevant to measurement must be identified. From them, the measurement goals may be derived, determining which information needs must be attained and the necessary measures for it. Goals, information needs and measures identification must follow the GQM paradigm [Basili *et al.* 1994]. However, since in OBA-MSI goals, information needs and measures identification should be aligned to the tools to be integrated, the established goals must be liable from measures that can be obtained from the tools. Therefore, this activity should be performed iteratively with *Identify Tools to provide Data and Services for Software Measurement* activity, ensuring that the defined goals, information needs and measured are in line with the tools to be integrated.

Identify Tools to provide Data and Services for Software Measurement is the activity in which the tools to be integrated are selected. Tools must be analyzed aiming at supporting goal monitoring by providing the required measures. Tools that the organization already uses (tools developed in-house or acquired) as well as others unused until then should be considered. Among the tools used by the organization, it should be considered the ones that support activities related to the measurement process, even if they are not tools dedicated to this purpose. OBA-MSI provides guidelines to assist the selection of tools able to supply data and services for software measurement.

Record the Integration Scenario consists in recording the results of the previous activities in the integration scenario, which contains: goals, information needs and measures to be addressed by the integration initiative, tools to be integrated, domains involved in the integration initiative (domains in which measurement will be applied), and activities of the measurement process that will be supported by integrated solution.

3.2 Integration Analysis

Once defined the integration scenario, the ***Integration Analysis*** phase can start. This phase has the same activities defined by OBA-SI (*Obtain Conceptual Models*, *Select Ontologies*, *Perform Vertical Mappings*, *Build the Integration Model*, and *Perform Horizontal Mappings*). However, there are two main differences.

First, in OBA-SI ontologies must be selected according to the domains involved in the integration. In OBA-MSI, apart from the selected tools or their domains, the ontologies to be used are RSMO and SMTO, since the process to be supported is software measurement. Thus, instead of selecting ontologies, in OBA-MSI one must identify RSMO and SMTO fragments relevant to the integration. If the domains to which measurement is to be applied are not addressed by RSMO, it should be extended.

Second, to accomplish integration at service and process layers, OBA-SI uses task ontologies to assign meaning to tool services and process activities. Mapping is used to establish a semantic equivalence between the service and the activity it supports. Since measurement is applied to several processes, OBA-MSI deals with the integration of tools supporting different processes and the integrated solution has always to support the same process: software measurement. In this sense, there are tool services that are not equivalent to activities of the software measurement process, because they are services of tools that support other processes. For instance, the service responsible for performing check-in of configuration items in a configuration management system can also collect data regarding configuration items (e.g., number of checked-in configuration items), however the service is not semantically equivalent to the Perform Measurement activity, although it supports this activity. In fact, the service is semantically equivalent to an activity of the configuration management process. Thus, in OBA-MSI, a mapping between a service and an activity of the software measurement process does not indicate semantic equivalence, but that the service supports performing the measurement activity.

Figure 4 illustrates the mapping between activities and services. First, activities of the organizational software measurement process are mapped to SMTO activities (solid arrows), indicating that there is semantic equivalence between them. This is done during the *Integration Requirements Elicitation* phase. For instance, in Figure 4, A1’ is mapped to A1, meaning that the A1’ activity of the organizational measurement process is semantically equivalent to the A1 activity of the SMTO measurement process. Then, tool services are mapped to SMTO activities (dashed arrows), meaning that the services support the corresponding activities. For instance, in Figure 4, S2 supports A1, and S4 supports A2. Finally, from the mappings it is possible to verify the support provided by tools services to the organizational measurement process. For example, service S2 supports activity A1. Since A1’ is equivalent to A1, then S2 supports A1’. Similarly, S4 supports A2, and A2’ is equivalent to A2, then S4 supports A2’. A3’ is equivalent to A3, however, there is no service supporting A3. Thus, to support A3 it is necessary to provide a service in the integrated solution.

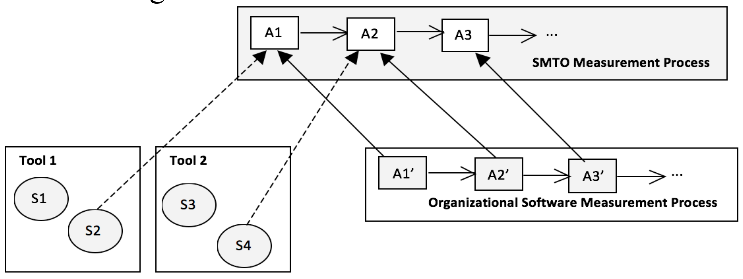


Figure 4. Mapping between services and activities.

4. Applying OBA-MSI in a Software Organization

OBA-MSI was used in an integration initiative to support the software measurement process at the Software Development Extension Laboratory (LEDS) [LEDS 2016], a software development organizational unit of a Brazilian Federal Institute. LEDS develops software by following Scrum [Cohn 2009] principles, and adopts tools to support project management and software development. The software manager reported the need of obtaining data for monitoring software projects and product quality. According to him, required data were scattered among tools, hindering access to them.

He also pointed out that getting additional data derived from data provided by the tools would be useful to support decision making. Due to space limitations, in this section we present only few results of the use of OBA-MSI at LEDS. Further information can be obtained in [Fonseca *et al.* 2016].

As described in the previous section, the first OBA-MSI activity (***Identify and Align Software Measurement Process***) concerns the alignment of the organization measurement process with SMTO. At the time OBA-MSI was applied, LEDS did not have a measurement process formally defined. Thus, following OBA-MSI, we analyzed its development process in order to identify activities related to software measurement. The results showed that in the context of the software development process there were some activities related to data collection. For instance, the development process had the *Estimate Project* activity, wherein values are estimated to project team velocity, user stories size and number of story points. The process also had the *Perform Continuous Integration* activity, in which source code is integrated and data regarding the source code is collected and stored. In order to make the measurement activity explicit, we split the last activity into *Perform Continuous Integration*, which deals with the continuous integration itself, and *Measure Source Code,* being devoted to perform measurements regarding source code.

After identifying the measurement-related activities, we defined the organizational measurement process considering the identified activities and SMTO. We started by mapping the measurement-related activities with SMTO activities. The mapping revealed that only activities related to the *Perform Measurements* activity could be clearly identified in the LEDS' development process. However, although not defined in the development process, activities related to *Plan Measurement* and *Analyze Measurements* were also performed at LEDS. For instance, when starting a project, the manager planned targets to be achieved, such as a desired value interval for source code duplications rate in the project. At the end of a sprint or of the project, he checked if the established targets were achieved.

Taking the three main SMTO activities as basis, we detailed each one of them considering the identified measurement-related activities or SMTO activities. Figure 5 shows the resulting measurement process. Activities in grey boxes are measurement-related activities that occurs in the context of the development process. The others were included based on SMTO. It is worth noticing that the measurement-related activities identified from the development process and included in the measurement process contributes to establish a direct connection between the LEDS' measurement and development processes, and makes explicit in which points of the development process measurements should occur.

Once the software measurement process was defined and aligned with SMTO, the ***Identify Goals, Information Needs and Measures relevant to the Organization*** activity was performed aiming to establish the goals to be monitored and the measures to be addressed by the integration initiative. As defined in OBA-MSI, this activity was performed iteratively with the selection of the tools to be integrated, aiming to ensure that identified measures could be obtained from the selected tools. Two business goals (BG) were identified: *Improve software projects management* and *Improve source code quality*. From these BGs, 7 measurement goals (MG) were derived (e.g., *Monitor sprint performance*), and 34 information needs (IN) and measures (ME) were identified (e.g., *Sprint User Story Conclusion Rate* and *Sprint Task Conclusion Rate*).

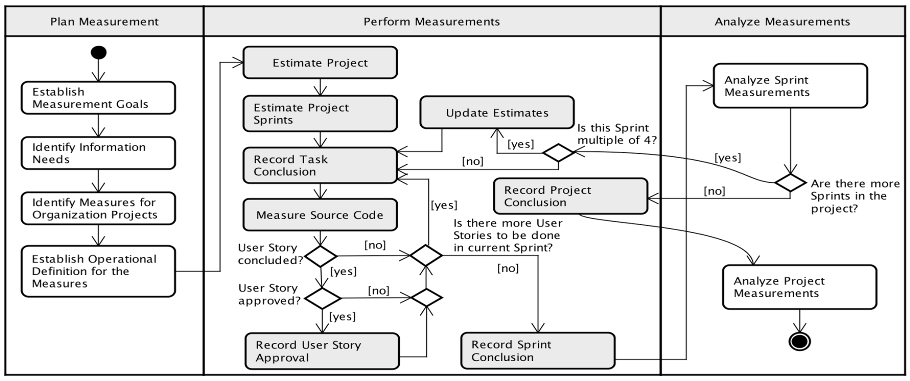

Plan Measurement
Perform Measurements
Analyze Measurements
Establish Measurement Goals
Identify Information Needs
Identify Measures for Organization Projects
Establish Operational Definition for the Measures
Estimate Project
Estimate Project Sprints
Record Task Conclusion
Measure Source Code
User Story concluded?
[no]
[yes]
User Story approved?
[no]
[yes]
Record User Story Approval
Update Estimates
[yes]
Is this Sprint multiple of 4?
[no]
[yes]
Is there more User Stories to be done in current Sprint?
[no]
Record Project Conclusion
Record Sprint Conclusion
Analyze Sprint Measurements
[yes]
[no]
Are there more Sprints in the project?
Analyze Project Measurements

Figure 5. Measurement Process defined from the LEDS Development Process and SMTO.

In the ***Identify Tools to provide Data and Services for Software Measurement*** activity, we followed OBA-MSI guidelines to select the tools to be integrated. Initially, we identified the tools used by LEDS for supporting the measurement-related activities. At this moment, Taiga[1] (a tool supporting agile project management) and SonarQube[2] (a tool supporting source code quality evaluation) were selected. Since none of these tools was developed to support the measurement process, a specific software measurement tool had to be selected. We chose SoMeSPC[3], a tool supporting software measurement and statistical process control, developed in the authors' research group for supporting activities of the software measurement process. After selecting the tools, the integration scenario is established.

Once the integration scenario was established, the ***Integration Analysis*** phase started. SMTO and RSMO were used for guiding semantic integration. Since the integration scenario includes aspects related to agile software processes (Scrum), it was necessary to extend RSMO by integrating concepts related to that domain. Then, the conceptual estructural models of Taiga, SonarQube and SoMeSPC were obtained by analyzing the tools and their documentation. After getting the tools' conceptual models, we used RSMO as basis to establish vertical mappings, assigning semantics to the tools' elements by relating them to RSMO elements. After vertical mappings, we built the integration model to enable integration at data layer. In order to address service and process layers, following OBA-MSI guidelines, we identified services available in the tools' APIs able to support SMTO measurement process activities. For instance, the services Measurement Registration, Source Code Registration and Sprint Registration, provided respectively by SoMeSPC, SonarQube and Taiga, are able to support the *Collect Data* activity (a sub-activity of *Perform Measurements* in SMTO). The use of services to tools communication regards to integration at the service layer, since services are used to exchange data.

Considering the mappings between services and activities of the SMTO measurement process, and the alignment of the organizational measurement process with SMTO (established when the measurement process was defined based on SMTO), the services were mapped to the activities of the LEDS' measurement process, indicating the activities by the tools' services. Thus, once tool integration is performed

[1] *http://taiga.io/*
[2] *http://www.sonarqube.org/*
[3] *http://github.com/nemo-ufes/SoMeSPC*

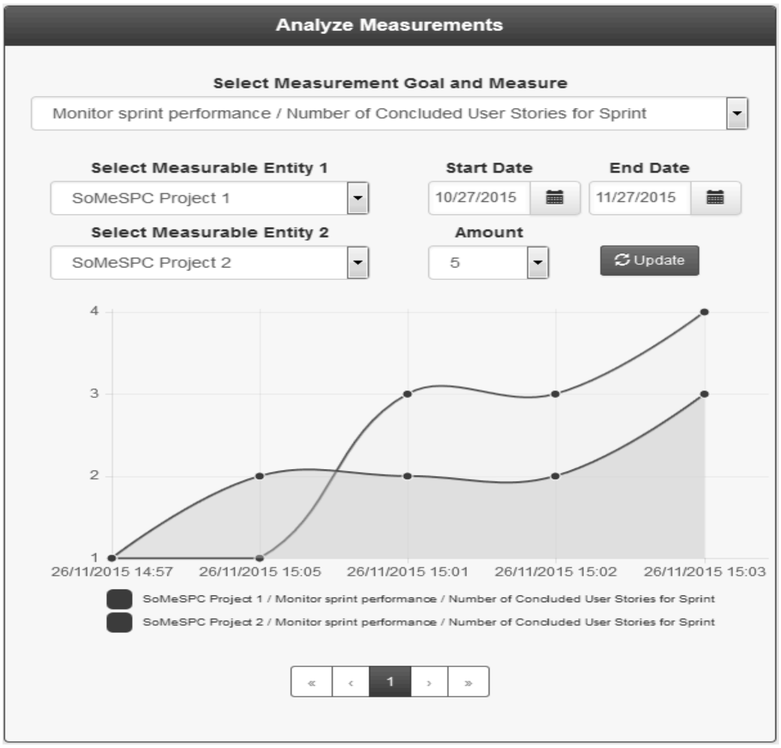
considering the identified services, an integrated support is provided for the organizational measurement process, covering also the integration at the process layer.

Integration design and implementation were done by the means of a mediator, which was developed based on the integration model and the services mapped to activities of the measurement process. The mediator consists of a Java web application developed as a SoMeSPC module extension. It is responsible for coordinating services in order to support all activities of the measurement process, considering a holistic view of it. Three main features are provided:

(i) A wizard for guiding the definition of Project Measurement Plans, as a result of the Plan Measurement activity. User is guided step-by-step from goal selection (each measurement goal is associated to information needs and to measures identified during *Integration Requirements Elicitation* phase) to the establishment of operational definition of measures. For each measure included in the Project Measurement Plan, a job is created to automatically collect data for the measure, considering the periodicity indicated in its operational definition.

(ii) A panel to manage (execute, pause or delete) the created jobs to collect data from the integrated tools.

(iii) A goal-based analysis feature that renders measured values into charts, supporting measurement analysis by providing useful information to monitor the established goals and to make decisions. Figure 6 illustrates selected data represented in a chart to support measurements analysis. In the figure, data related to the number of concluded user stories for sprints of two projects are plotted, providing information regarding the sprint performance.


Analyze Measurements

Select Measurement Goal and Measure: Monitor sprint performance / Number of Concluded User Stories for Sprint

Select Measurable Entity 1: SoMeSPC Project 1

Start Date: 10/27/2015

End Date: 11/27/2015

Select Measurable Entity 2: SoMeSPC Project 2

Amount: 5

Update
4
3
2
1
26/11/2015 14:57
26/11/2015 15:05
26/11/2015 15:01
26/11/2015 15:02
26/11/2015 15:03
SoMeSPC Project 1 / Monitor sprint performance / Number of Concluded User Stories for Sprint
SoMeSPC Project 2 / Monitor sprint performance / Number of Concluded User Stories for Sprint
«
‹
1
›
»

Figure 6 – Goal-based analysis feature provided by the mediator.

The integrated solution was made available for the organization usage and some weeks later we applied a survey in order to collect feedback from the team and the manager. The survey consisted of 12 questions regarding adequacy and utility of the features provided, and the benefits obtained from the use of the integrated set when compared to the isolated use of the tools. Also, an interview was conducted with the LEDS' manager. The results showed that the features were suitable and useful to support the measurement process at LEDS. The participants commented that make the measurement process explicit helped them to understand the process as a whole and measure what is really important to the organization. Also, all the participants stated that the integrated solution provides more benefits than using the tools in isolation.

5. Related Work

In the literature, there are some tool integration initiatives that support software measurement. We carried out a systematic investigation and identified 12 integration initiatives involving tool integration to support software measurement (see [Fonseca *et al.* 2015a, 2015b]). Most of the investigated initiatives integrate code-related tools. Consequently, most of the measures addressed in the initiatives are code-related measures (e.g., cyclomatic complexity, number of methods). The predominance of code-related tools might be due to the fact that code-related measures are prone to automatic collection. In the integration initiative we conducted at LEDS, we have integrated, besides a code-related tool, tools supporting project management and software measurement.

None of the found initiatives addresses integration at process layer. When applying OBA-MSI, the integration at process layer is addressed by using a task ontology (SMTO) as basis to assign semantics to services and process activities. Besides, the starting point to the integration in OBA-MSI is the software measurement process to be supported, stimulating integration at process layer.

Only one of the found initiatives is concerned with semantic aspects (SOFAS [Ghezzi and Gall 2011]). However, although SOFAS uses ontologies during integration, two issues can be highlighted: (i) SOFAS applies ontologies related to the tools domain (issue tracking, version control and source code ontologies), while when applying OBA-MSI, the integration initiative uses software measurement ontologies (task and domain ontologies); (ii) the ontologies used by SOFAS are considered lightweight ontologies whose purpose is guaranteeing desirable computational properties [Guizzardi 2005], while ontologies used in OBA-MSI are reference ontologies, whose purpose is to make the best possible description of the domain in reality, regardless technological issues [Guizzardi 2005].

None of the investigated initiatives presented the method followed to perform the integration. Thus, we presumed that they have used ad-hoc approaches for integrating the tools. Not using a systematic approach for performing the integration 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 [Nardi *et al.* 2013b]. In this sense, OBA-MSI is a systematic approach that guides integration through the use of ontologies to assign semantics to the elements involved in the integration.

We did not found any proposal of a systematic approach to carry out tool integration aiming at supporting the software measurement process. There are some systematic general integration approaches, such as [Yan et al. 2008] and [Liu et al. 2012]. However, both the approaches do not address integration at process layer neither at semantic level.

6. Final Considerations

Software measurement is an important practice that provides useful information for decision making at organizational and project levels. In order to be appropriately performed and to provide the expected benefits, software measurement should be based on organizational and project goals [Florac and Carleton 1997].

The efficiency of software measurement is strongly related to the support provided by software tools [Dumke and Ebert 2007]. Although some tools used by organizations are not devoted for supporting the measurement process, they collect and store data useful for decision making and need to be integrated to support the measurement process. Tool integration is not an easy task [Themistocleous et al. 2004]. The heterogeneity of systems is the major problem faced during an integration initiative because each tool has its own data and process models and is implemented in an independent way [Izza 2009]. This leads to several conflicts, mainly related to the semantics of data and services shared between the tools [Pokraev 2009].

Ontologies are considered an important means to achieve semantic integration since they provide formal specifications for shared conceptualizations [Nardi *et al.* 2013a]. They can be used to establish a common understanding of the universe of discourse, serving as an interlingua for communication between systems and avoiding semantic problems [Calhau and Falbo 2010].

This paper presented OBA-MSI, an Ontology-Based Approach for Measurement Systems Integration. OBA-MSI helps organizations to establish an appropriated measurement process and tool support by using software measurement reference ontologies. OBA-MSI extends OBA-SI focusing on the *Integration Requirements Elicitation* phase and uses RSMO and SMTO ontologies to aid semantic integration. OBA-MSI considers a holistic view of the software measurement process by using SMTO as basis and by requiring the organizational software measurement process to be aligned with it. OBA-MSI treats measurement planning by following a goal-based approach. Besides, it allows for integration at data, service and process layers.

OBA-MSI was applied to integrate tools to support the measurement process in a software development organization in order to help software projects monitoring and software quality improving. Tools supporting project management, code analysis and software measurement were integrated by using a mediator. Feedback provided by the organization's team and manager showed that the measurement process was better supported by using the integrated solution than using the tools in isolation. Besides, by applying OBA-MSI, the organizational measurement process was made explicit and integrated to the development process.

The main contributions of this work are: (i) the Ontology-Based Approach for Measurement Systems Integration (OBA-MSI); (ii) the systematic mapping describing the panorama of integration initiatives for supporting software measurement [Fonseca *et al.* 2015a]; (iii) the systematic literature review that provides a deeper investigation into integration initiatives for supporting software measurement [Fonseca *et al.* 2015b]; and the integrated set of tools provided to LEDS aiming to support its software measurement process and that can also be used by other organizations.

The integration initiative at LEDS was conducted by one of the researchers involved in the OBA-MSI development. As future work, we plan to apply OBA-MSI to integrate tools without the researchers intervention. By doing that, we can evaluate the viability of other people use the proposed approach. We also intend to run a new survey to get feedback from LEDS team and manager after a longer time using the integrated set. We also plan: (i) use core measurement ontologies to define an approach to integrate measurement tools in any domain, and (ii) explore the use of the integrated set of tools implemented to LEDS as a tool to aid software measurement teaching.

Acknowledgement

This research is funded by the Brazilian Research Funding Agency CNPq (Processes 485368/2013-7 and 461777/2014-2). The authors acknowledge the Software Development Extension Laboratory (LEDS) for the opportunity of applying and evaluating OBA-MSI.

References

Barcellos, M. P. and Falbo, R. A. (2013). A Software Measurement Task Ontology. In *Proc. of the 28th Annual ACM Symposium on Applied Computing*. ACM Press, p. 311–318.

Barcellos, M. P., Falbo, R. A. and Rocha, A. R. (2013). A Strategy for Preparing Software Organizations for Statistical Process Control. *Journal of the Brazilian Computer Society*, v. 19, n. 4, p. 445–473.

Basili, V. R., Caldiera, G. and Rombach, H. D. (1994). Goal Question Metric Approach. *Encyclopedia of Software Engineering*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Calhau, R. F. and Falbo, R. A. (2010). An Ontology-Based Approach for Semantic Integration. In *Proc. of the 14th IEEE International Enterprise Distributed Object Computing Conference*. IEEE, p. 111–120.

Cohn, M. (2009). *Succeeding with Agile: Software Development Using Scrum*. Addison-Wesley Professional.

Dumke, R. and Ebert, C. (2007). *Software Measurement: Establish - Extract - Evaluate - Execute*. Berlin, Heidelberg: Springer Berlin Heidelberg.

Florac, W. a. and Carleton, A. D. (1997). *Measuring the Software Process: Statistical Process Control for Software Process Improvement*. Boston, USA: Addison Wesley.

Fonseca, V. S., Barcellos, M. P. and Falbo, R. D. A. (2016). Integrating Tools to Support Software Measurement. In Proc. of the *15th Brazilian Simposium on Software Quality (SBQS 2016),* Maceió, Brazil.

Fonseca, V. S., Barcellos, M. P. and Falbo, R. de A. (2015a). Integration of Software Measurement Supporting Tools: A Mapping Study. In Proc. of the *27th International Conf. on Software Engineering and Knowledge Engineering (SEKE)*. Knowledge Systems Institute, p. 516–521.

Fonseca, V. S., Barcellos, M. P. and Falbo, R. de A. (2015b). Tools Integration for

Supporting Software Measurement : A Systematic Literature Review. *iSYS - Information Systems Brazilian Journal*, v. 8, n. 4, p. 80–108.

Ghezzi, G. and Gall, H. C. (2011). SOFAS: A Lightweight Architecture for Software Analysis as a Service. *2011 Ninth Working IEEE/IFIP Conference on Software Architecture*, p. 93–102.

Guizzardi, G. (2005). Ontological Foundations for Structural Conceptual Models, ISBN 90-75176-81-3, Universal Press, The Netherlands, 2005.

ISO/IEC (2007). *IEEE Standard Adoption of ISO/IEC 15939:2007—Systems and Software Engineering—Measurement Process*.

Izza, S. (2009). Integration of Industrial Information Systems: from Syntactic to Semantic Integration Approaches. *Enterprise Information Systems*, v. 3, n. 1, p. 1–57.

LEDS (2016). Software Development Extension Laboratory. http://leds.sr.ifes.edu.br/, [accessed on Mar 14].

Liu, C., Wang, J., Wen, Y. and Han, Y. (jun 2012). A Unified Data and Service Integration Approach for Dynamic Business Collaboration. In *2012 IEEE First International Conference on Services Economics*. IEEE, p. 54–61.

McGarry, J., Card, D., Jones, C., et al. (2002). *Practical Software Measurement: Objective information for decision makers*. Boston, USA: Addison Wesley.

Nardi, J. C., Falbo, R. A. and Almeida, J. P. A. (2013a). A Panorama of the Semantic EAI Initiatives and the Adoption of Ontologies by these Initiatives. In: *IWEI 2013, LNBIP 144*. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer Berlin Heidelberg. v. 144p. 198–211.

Nardi, J. C., Falbo, R. A. and Almeida, J. P. A. (2013b). Foundational Ontologies for Semantic Integration in EAI: A Systematic Literature Review. In: *I3E 2013, IFIP AICT 399*. IFIP Advances in Information and Communication Technology. Berlin, Heidelberg: Springer Berlin Heidelberg. v. 399p. 238–249.

Pokraev, S. (2009). Model-Driven Semantic Integration of Service-Oriented Applications. University of Twente.

Solingen, R. and Berghout, E. (1999). The Goal/Question/Metric Method: A Practical Guide for Quality Improvement of Software Development. *New York, McCraw-Hill Publishers*.

Themistocleous, M., Irani, Z. and Love, P. E. D. (2004). Evaluating the Integration of Supply Chain Information Systems: A Case Study. *European Journal of Operational Research*, n. 159, p. 393–405.

Wegner, P. (mar 1996). Interoperability. *ACM Computing Surveys*, v. 28, n. 1, p. 285–287.

Yan, W. J., Tan, P. S. and Lee, E. W. (jul 2008). A Web Services-enabled B2B Integration Approach for SMEs. In *2008 6th IEEE International Conference on Industrial Informatics*. IEEE, p. 774–779.