Software Testing Processes in ISO Standards: How to Harmonize Them?

Fabiano B. Ruy[1,3], Érica F. Souza[2], Ricardo A. Falbo[3], Monalessa P. Barcellos[3]

[1]Informatics Department – Federal Institute of Espírito Santo (IFES), Campus Serra, Serra, ES, Brazil

[2]Computer Department – Federal Technological University of Paraná (UTFPR), Cornélio Procópio, PR, Brazil

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

{fabianoruy, falbo, monalessa}@inf.ufes.br, ericasouza@utfpr.edu.br

Abstract. *Software organizations usually adopt quality standards for improving their testing processes. ISO provides different standards addressing the testing process, such as ISO/IEC 12207, ISO/IEC 29110 and ISO/IEC 29119. However, these standards are not properly aligned and, when used in combination, can give rise to conceptual inconsistencies and divergences. This paper presents an initiative harmonizing ISO testing processes extracted from these standards. Two ontologies were used in such initiative: a Software Process Ontology for harmonizing the standards' structure, and a Reference Ontology on Software Testing (ROoST) for harmonizing standards' content.*

1. Introduction

Software Verification & Validation (V&V) activities intend to ensure, respectively, that a software product is being built in conformance with its specification, and that it satisfies its intended use and user needs [IEEE 2004]. Software testing is the dynamic V&V of the behavior of a program against the expected one, considering a finite set of test cases [Bourque and Fairley 2014]. It is a fundamental process for software organizations, since it allows achieving quality software products. To be effective, testing activities should be integrated into a well-defined and controlled process. The importance of testing processes is widely recognized, and there is a growing concern in how to improve the accomplishment of this process [TMMi Foundation 2012].

To define and improve their testing processes, software organizations can adopt testing-related standards that support better defining the activities to be performed, artifacts to be produced, roles involved, and other elements of this process. These standards are used to guide software organizations efforts towards quality software testing processes. In this context, the *International Organization for Standardization* (ISO) has a set of standards covering the testing domain, such as ISO/IEC 12207 [ISO/IEC 2008], ISO/IEC 29110 [ISO/IEC 2011] and ISO/IEC 29119 [ISO/IEC 2013]. ISO/IEC 12207 and ISO/IEC 29110 are broader standards in the sense that they provide a comprehensive description of software life cycle processes, including recommended testing practices. On the other hand, ISO/IEC 29119 is specific for software testing. Its purpose is to define a generic process model for software testing that can be used by any organization when performing any form of software testing [ISO/IEC 2013].

The combined use of standards can help software organizations to define and improve their testing processes. When an organization adopts related standards, it can exploit synergies between them, overcoming the weakness of a single standard by the strengths of the others [Jeners et al. 2013]. Regarding the testing domain, for example, while ISO/IEC 12207 and 29110 are more general, ISO/IEC 29119 provides richer details concerning testing activities and work products. While ISO/IEC 29119 focuses on testing, the other two offer a better view on how testing activities interrelate with the other development processes. Thus, the process team can select the portions of each standard that better address the organization's goals for defining its processes.

However, implementing a process adherent to multiple standards is not an easy task. Besides the well-known difficulties on implementing each standard, there is a significant effort on considering the portions from different standards dealing with the same aspects. Most of the standards are created independently, by different groups or organizations, defining their own scope, structure, abstraction level and terminology [Biffl et al. 2006], and not necessarily sharing the same semantics. Even standards developed by the same standardization organization, attempting to share a similar conceptual base (e.g. ISO/IEC 24744 metamodel intends to be a common basis for ISO Software Engineering standards [ISO/IEC 2007]), are not perfectly aligned and can present structural and conceptual divergences. The ISO itself has started an initiative for harmonizing its own standards [Henderson-Sellers et al. 2014].

Among the main identified problems regarding the combined use of multiple standards, there are [Yoo et al. 2006] [Pardo et al. 2013] [Jeners et al. 2013]: diversity in the representation structure; diversity in the adopted vocabulary (in terms and meanings); differences in elements granularity; and divergent presentation (regarding language, abstraction level, and subjectivity). Hence, when multiple standards are used in combination, some harmonization effort should be considered. By harmonizing the standards, they can be analyzed and compared, similarities can be identified and divergences solved, providing an integrated view of the target standards. Thus, organizations can reduce costs and efforts on the standards deployment. These problems can be associated to the structure and content of the involved standards. Structural issues arise when standards are organized in different ways. They can conflict, for example, by using different notions, hard to be directly correlated; or worst, by applying the same notions differently. Content problems, in turn, regard the knowledge represented by the standards, which, again, can be described in different ways. Different terms can be used to refer to equivalent content, making difficult to correlate them.

This paper discusses how test-related processes and activities provided by distinct standards can be harmonized with the support of ontologies and conceptual models. It focuses on testing processes extracted from the standards ISO/IEC 12207, 29110 and 29119, identifying some structural and content problems and discussing how they can be addressed. The remainder of the text is organized as follows. Section 2 presents a background on Software Testing and related ISO standards, and introduces some harmonization concepts, perspectives and techniques. Section 3 discusses structural aspects regarding the selected ISO standards and how they can be addressed. Section 4 focuses on content aspects, presenting a content mapping and discussing some inconsistencies and divergences. Section 5 addresses related works, and Section 6 concludes the paper.

2. Background

To achieve quality software products, it is essential to perform Verification & Validation (V&V) activities throughout the software development process. V&V activities can be static and dynamic. The dynamic activities require the execution of a program, while static activities do not. Static V&V are typically done by means of technical reviews and inspections. Dynamic V&V, the focus of this study, are done by means of testing [Mathur 2012].

Ideally, testing activities should be defined and organized as a well-defined process. Although each organization may introduce particularities in its testing process, in general, the following activities should be considered: Test Planning, Test Design, Test Execution and Test Result Analysis [Bourque and Fairley 2014]. Key aspects of Test Planning include, among others, coordination of personnel, management of available test facilities and test environment, scheduling testing activities and planning for possible undesirable outcomes. Test Case Design aims at designing the test cases and test procedures. During Test Execution, test cases are executed, producing actual results. In Test Result Analysis, test results are evaluated to determine whether (or not) tests have been successful in identifying defects.

There are several standards that can support testing process improvement. Among the ISO standards covering the testing domain, there are: ISO/IEC 12207 [ISO/IEC 2008], ISO/IEC 29110 [ISO/IEC 2011] and ISO/IEC 29119 [ISO/IEC 2013].

ISO/IEC 12207 provides a comprehensive set of life cycle processes, activities and tasks for software. In relation to software testing, ISO/IEC 12207 provides activities and tasks in many software specific processes. The Software Construction Process includes activities to elaborate test procedures and test data for unit testing. In the Software Integration Process, the main testing activities are to develop, document and evaluate a set of test cases and test procedures for conducting qualification tests. The Software Qualification Testing Process purpose is to confirm that the integrated software product meets its defined requirements. This process may be used in the Verification and Validation Processes. Finally, Software Verification Process confirms that each software work product properly reflects the specified requirements, while Validation confirms that the requirements for an intended use are fulfilled.

ISO/IEC 29110 is to be used by very small organizations for establishing processes to implement development approaches considering some activities, including software testing. The main testing tasks in this standard are established in the following activities: Software Architectural and Detailed Design (includes tasks related to the design of test cases and test procedures), Software Construction (includes unit testing tasks, from its design to execution), and Software Integration and Tests (involves setting, executing and updating test cases and test procedures).

Unlike the other two standards, ISO/IEC 29119 is specific for software testing. Its purpose is to define a set of standards for software testing that can be used by any organization when performing any form of software testing. It is divided in four parts: Part 1 - Concepts and definitions, Part 2 - Test processes, Part 3 - Test documentation, and Part 4 - Test techniques. The Part 1 introduces the main concepts of software testing. Part 2 presents several test processes, organized in a model comprising three layers: Organizational Test Process, Test Management Processes and Dynamic Test

Processes. Each layer describes processes composed of sub-processes. Part 3 describes test documents that are output of the test processes specified in Part 2. Finally, Part 4 defines test design techniques that can be used during the Test Design and Implementation process defined in Part 2.

Different standards on the same area are often applied together. However, this combined adoption requires some harmonization. Standards Harmonization is not a recent topic in Software Engineering. Since the 1990's diverse initiatives have been conducted to alleviate the effects of standards incompatibilities and to present a more aligned view of them. Examples are mappings comparing specific standards (such as ISO 9001 and CMM/CMMI [Paulk 1993]); development of conceptual models representing a domain comprising a set of standards [García et al. 2006] [Falbo & Bertollo 2009]; standards providing some alignment or mappings to other related standards (such as MR-MPS.BR [SOFTEX 2016] does to CMMI-DEV [SEI 2010], and ISO/IEC 29110 does to ISO/IEC 12207); and a number of approaches and techniques aiming to homogenize, compare, map, integrate and harmonize standards [Ferchichi et al. 2008] [Jeners et al. 2013] [Pardo et al. 2013] [Henderson-Sellers et al. 2014].

These initiatives vary in the domains and standards chosen, harmonization perspectives, levels of detail, techniques applied, among others. Regarding the harmonization perspective, some initiatives focus on the standards ***structure***, analyzing and making correspondences between structural elements only, i.e. those used to organize/categorize the standards contents (such as *process*, *activity* and *task* in ISO/IEC 29110); and some focus on the standards' ***contents***, dealing with the represented knowledge (such as the *Software Implementation* process, the *Software Integration and Test* activity, and the *Perform Software Tests* task in ISO/IEC 29110). While the former presents a way for understanding the structural similarities and differences between distinct standards, the second deals with the more specific knowledge, used to describe the processes that are the target of the standards.

Several techniques have been applied for harmonizing standards [Pardo et al. 2013]. **Homogenization** is used for adapting the standards to a pre-defined structure. This technique is related to the *structural* perspective, and can be used to support other techniques. **Comparison** analyzes standards or their processes looking for similarities and differences, while **Mapping** focuses on comparing the elements (such as activities, work products and roles) and establishing links between them. Comparison and Mapping can be used in both *structural* and *content* perspectives. Finally, **Integration** consists in defining a new (integrated) artifact unifying the selected elements of different standards. It can be applied to both standards' *structure* and *content*. The more elaborated harmonization approaches often apply a set of these techniques [Ferchichi et al. 2008] [Jeners et al. 2013] [Pardo et al. 2013], e.g., first homogenizing the involved standards to the same format, next mapping them to identify related elements, and finally deriving an integrated artifact adherent to all involved standards.

Since these tasks frequently involve knowledge representation, some approaches make use of conceptual models to support the harmonization efforts and for better dealing with the standards' knowledge [Jeners et al. 2013] [Pardo et al. 2015]. The effective application of the techniques requires understanding the involved standards and, mainly, the semantics underlying them and their domains. It is necessary to deal with the elements meanings, and not only with the terms used to name them. In this

context, a semantic referential can be used for confirming the similarities and for guiding a suitable domain-grounded solution when the standards conflict. Ontologies have been frequently used in this quest [Henderson-Sellers et al. 2014] [Pardo et al. 2015]. An ontology is a formal, explicit specification of a shared conceptualization [Studer et al. 1998]. In the context of standards harmonization, ontologies apply by offering a well-agreed structure for homogenizing standards and classifying their elements. Ontologies can provide concepts and relations specialized to the target domain of harmonization, and thus can be used to solve inconsistencies and terminological conflicts, bringing about benefits such as the provision of precise and clear definitions and a more straightforward representation of processes and standards [Pardo et al. 2013].

In the next sections, we discuss structural and content harmonization, including problems and conflicts we have found in the aforementioned test-related ISO Standards. These issues are addressed by applying harmonization techniques supported by an ontological framework.

3. Software Testing in ISO Standards: Structure Harmonization

To understand the structural differences between the standards, we have developed a structural model for each one. A Standard Structural Model (SSM) is a conceptual model extracted from a standard, considering only its structural elements, i.e. those elements used to describe processes in the standards (such as *process*, *activity* and *artifact*), disregarding their contents (e.g., the specific processes, activities and artifacts being described in the standard). Once the SSMs have been developed, we performed a structural analysis to identify similarities and differences between them.

ISO standards, in general, decompose their *work units* in three granularity levels: process, activity and task. ISO/IEC 12207 provides definitions for such types of work units that are used by the other two standards. *Process* is defined as "a set of interrelated or interacting activities which transforms inputs into outputs". *Activity* is defined as "a set of cohesive tasks of a process". Finally, *Task* is defined as a "requirement, recommendation, or permissible action, intended to contribute to the achievement of one or more outcomes of a process". In addition, a process aims to achieve a *purpose*, and shall result in *outcomes*. An outcome statement can describe: the production of an artifact, a significant change in state, or the meeting of specified constraints, such as requirements, goals, etc. [ISO/IEC 2008]. Moreover, the three considered standards deal with stakeholders and with work products in similar ways: stakeholders participate in work units; while work units produce (output) and use (input) work products. *Stakeholder* is defined as an "individual or organization having a right, share, claim or interest in a system or in its possession of characteristics that meet their needs and expectations" [ISO/IEC 2008]. *Work product* is defined as an "artifact associated with the execution of a process" [ISO/IEC 2011].

Figure 1 shows the SSMs of the three considered standards. In the SSMs, concepts in yellow and relations in black are those common to the three standards. Concepts and relations in red are those that are specific to a standard, and thus they cannot be harmonized. It is worth noting that concepts representing *stakeholders* and *work products* are included in the SSMs even not being explicitly defined in the standards' structure (as it is the case of ISO/IEC 12207), but addressed in their texts.

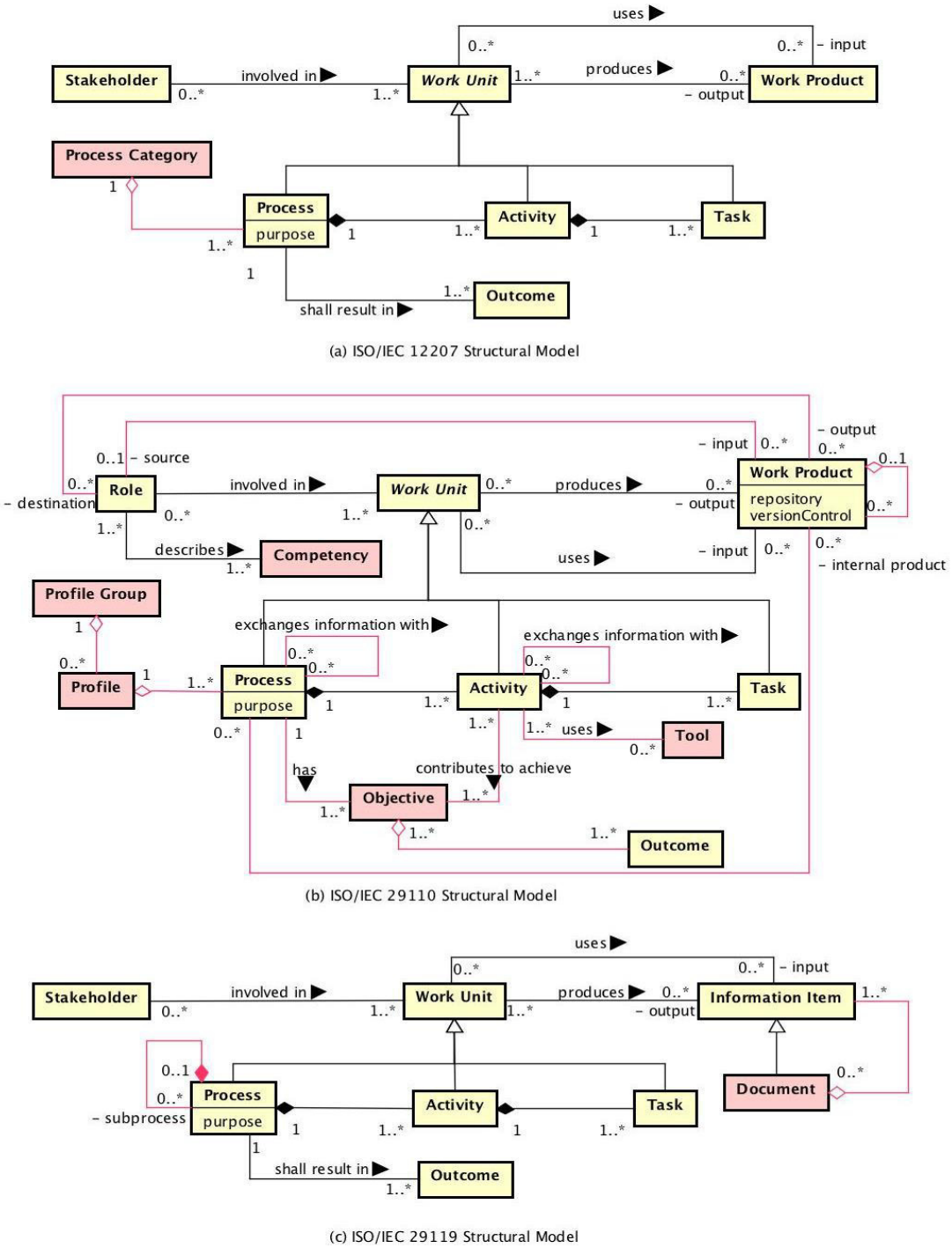

Figure 1. Standards Structural Models

As Figure 1 shows, ISO/IEC 12207 has the simplest structure. Besides the concepts and relations that are common to the other standards, it groups processes into *Process Categories*. ISO/IEC 29119 is structurally strongly aligned to ISO/IEC 12207. The main differences are admitting processes to be decomposed into sub-processes, and considering documents as a composite work product (named *Information Item* in ISO/IEC 29119).

ISO/IEC 29110 is the most detailed standard in terms of structure. Processes are grouped into *Profiles* that, in turn, are grouped in *Profile Groups*. Information regarding

the way processes (and also activities) exchange information with other processes (and activities) is also present in this standard. Stakeholders (named *Roles* in this standard) are described in terms of *Competencies*, and can be the *source* or the *destination* of work products. Moreover, ISO/IEC 29110 points out products generated and consumed by a process (said *internal products*), as well as describes how some work products are decomposed into other work products. Finally, this standard also points out *Tools* that can be used to support activities, and how the activities contribute to achieve the process objectives. An *Objective* is a specific goal to ensure the accomplishment of the process purpose, and includes the outcomes related to the objective.

Although at first glance the SSMs of the three standards share several commonalities (concepts in yellow and relations in black in Figure 1), when looking deeper, there are still differences even where the standards seem to converge. Next, we present some problems we detected when analyzing the portions of these standards related to software testing:

(1) Vocabulary-related Problems: Although both ISO/IEC 29110 and 29119 refer to 12207 as being their structural basis, different terms are used to designate the same concept. For instance, ISO/IEC 29110 uses "Role", while 12207 and 29119 use "Stakeholder" to refer to the same notion. ISO/IEC 29119 uses "Information Item", while 12207 and 29110 use "Work Product" to refer to the notion of artifacts produced and consumed by work units. Organizations using these standards together must identify those "synonyms" to link the corresponding notions.

(2) Work Unit Granularity: Although the three standards adopt a structure in which processes are composed of activities that are composed of tasks, what is considered a process, an activity or a task varies. For example, what is considered a *process* in ISO/IEC 29119 can be defined as an *activity* in 12207 or 29110. ISO/IEC 29119 is devoted to software testing, and describes processes composed of sub-process (e.g., the Test Management Process is composed of Test Planning, Test Monitoring & Control, and Test Completion Processes), which are then decomposed into testing activities and tasks. ISO/IEC 12207 defines a Software Qualification Testing Process, but other processes such as Software Construction Process and Software Integration Process clearly include activities and/or tasks related to test. In ISO/IEC 29110, test activities and tasks are part of the Software Implementation Process. Identifying the correspondences between work units described at different levels in different standards is a challenge, since we need to analyze each work unit without considering the work unit granularity defined by the standards.

(3) Work Products Descriptions: Different standards define work products in different levels of detail. While ISO/IEC 12207 does not define clearly the work products (they are only mentioned in the text, often subjectively), ISO/IEC 29110 and 29119 (Part 3 is devoted to test documentation) provide detailed descriptions of work products, describing even the information items that they must include. Again, identifying correspondences between work products described at different levels of details is a challenge, since the details provided, many times, are not enough to allow establishing proper correspondences.

(4) Stakeholders Descriptions: Analogously to work products, ISO/IEC 29110 and 29119 define explicitly the stakeholders/roles involved in the activities and tasks, while 12207 is almost always neutral in relation to this (most of the times just mentioning a generic process "implementer" role).

(5) Outcomes Descriptions: Both ISO/IEC 12207 and 29119 define outcomes related to processes, while 29110 defines process objectives that are related to outcomes. Although 29110 adopts a little different view, the outcomes considered in this standard are exactly the same of the ones considered in 12207, making a direct link between these two standards. The main problem in this case is to find which outcomes refer to the testing process. This problem is amplified in 12207, since testing activities and tasks are spread out in several processes. In the case of 29119, once it is difficult to establish correspondences between its work units and the ones from 12207, it is hard to establish correspondences between the outcomes. Moreover, outcomes refer to very different things (the production of an artifact, a significant change in state, or the meeting of specified constraints), so it becomes even harder to compare outcomes from these two standards.

By developing the SSMs, it becomes easier to compare and map standards' concepts and relations. However, we must still deal with divergences between the standards, and a direct correspondence could be not precise, or even possible. Instead of just linking the related structural elements, we advocate in favor of using an ontology as *interlingua*. This allows us to use a semantic referential to which we can link each SSM element. The ontology provides the domain grounding, independently of any specific standard. Thus, structural inconsistencies and divergences can appear more explicitly and be solved by following the semantic referential. Since all structural elements are related to the corresponding ontology concepts, the SSMs can be homogenized according to the ontology, being more precisely compared. Moreover, later, during content harmonization, this referential guides the mapping by allowing only elements with similar semantics to be compared.

Figure 2 presents a fragment of the Software Process Ontology (SPO) of SEON[1] (Software Engineering Ontology Network) [Ruy et al. 2016], useful for the scope of our harmonization effort (i.e., *work units*, *work products* and *stakeholders* and how they relate). We used this fragment for harmonizing the structure of the three ISO standards considered. It is important to highlight that we did not consider outcomes in the scope of our harmonization effort, because *outcomes* refer to very different things, being hard to establish a meaning for this concept.

According to SPO, *processes* are actions performed by software organizations. There are two types of processes: *General process* represents overall process performed in a specific project or organization, which is necessarily composed of *specific processes*. Specific processes are composed of *activities*. Analogously, an activity can be simple or composite. A *composite activity* is composed of other activities, while a *simple activity* is an atomic action that cannot be decomposed into smaller activities. Activities produce and use *artifacts*. Artifacts can be simple or composite. *Composite artifacts* are composed of other artifacts; *simple artifacts* are atomic objects. Moreover, regarding its nature, an artifact can be a *software product*, a *software item*, a *model*, an *information item* or a *document*. *Stakeholders* participate in activities.

[1] SEON is available at https://nemo.inf.ufes.br/seon/.

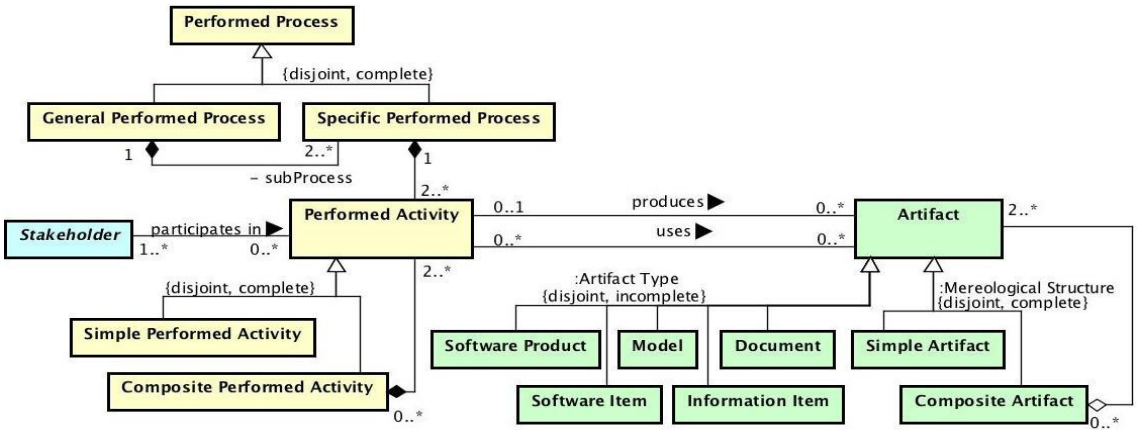

Figure 2. The SPO fragment used for harmonizing the Standards Structural Models.

It is important to say that SPO is grounded in the Unified Foundational Ontology – UFO [Guizzardi et al. 2008]. According to UFO, processes and activities (shown in yellow in Figure 2) are *actions*; artifacts (shown in green in Figure 2) are *objects*; and stakeholders (shown in blue in Figure 2) are *agents*. Based on this foundation, in Table 1, we establish admissible mappings between the standards' elements and the ontology' concepts, clarifying some notions used in the SSMs.

Table 1. Admissible Structural Mappings between Standards' Elements and Ontology's Concepts.

Ontology Concept	ISO 12207 Element	ISO 29110 Element	ISO 29119 Element
Performed Process (action)	Process (action) Activity (action)	Process (action) Activity (action)	Process (action)
Performed Activity (action)	Process (action) Activity (action) Task (action)	Activity (action) Task (action)	Process (action) Activity (action) Task (action)
Artifact (object)	Artifact (object)	Product (object)	Information Item (object) Document (object)
Stakeholder (agent)	Stakeholder (agent)	Role (agent)	Stakeholder (agent)

For example, concerning work units, we can see equivalent elements with distinct classification. "Perform the defined tests" is considered a *task* in ISO/IEC 29110 (SI 5.4); both an *activity* (7.1.7.3.1) and a *task* (7.1.7.3.1.1) in ISO/IEC 12207; and a *process* (8.4) or an *activity* (8.4 TE1) in ISO/IEC 29119. With the ontology, each structural element can be linked to the proper concept according to its real meaning. The alignment of the SSMs by means of an ontology is a fundamental step in harmonizing these standards since it "translates" each standard to the same semantic interlingua, and defines which types of elements can be compared during the content harmonization, as discussed in next section.

4. Software Testing in ISO Standards: Content Harmonization

Regarding the standards' contents, ISO/IEC 29119, being the one specific for software testing, is the most complete and detailed. It establishes several test-related processes, organized in three layers. The first layer contains only one process: the Organizational Test Process, which deals with organizational test policies, strategies, processes, procedures and other assets. The second layer, Test Management Processes, defines three processes that cover test management, namely: Test Planning Process, Test

Monitoring and Control Process, and Test Completion Process. Finally, the last layer, Dynamic Test Processes, defines generic processes for performing dynamic testing, namely: Test Design and Implementation Process, Test Environment Set-Up and Maintenance Process, Test Execution Process, Test Incident Reporting Process.

ISO/IEC 12207 is the most general, and is organized in terms of System Context Processes (including Agreement, Organizational, Project and Technical Processes) and Software Specific Processes (including Software Implementation, Support and Reuse Processes). The Life Cycle Model Management Process aims to define, maintain, and assure availability of policies, life cycle processes, life cycle models, and procedures for use by the organization. In this sense, we can say that it is responsible to deal with organizational test policies, strategies, processes and procedures, like the Organizational Test Process of ISO/IEC 29119. The Infrastructure Management Process aims to provide the enabling infrastructure and services to projects to support organization and project objectives throughout the life cycle. Thus, it covers the Test Environment Set-Up and Maintenance Process of ISO/IEC 29119. The Project Planning and the Project Assessment and Control Processes addresses project planning, activation, monitoring, control, assessment and closure, and cover the Test Management Processes of ISO/IEC 29119. The remainder processes of ISO/IEC 29119 are covered by activities and tasks spread out in different Software Specific Processes of ISO/IEC 12207, including the following (as discussed in Section 2): Software Construction, Integration, Qualification Testing, Verification and Validation Processes.

ISO/IEC 29110 is also a general standard, but simpler, since it is devoted to very small organizations. It does not cover organizational aspects, neither aspects related to infrastructure. It has only two processes: Project Management Process and Software Implementation Process. The Project Management Process comprises activities to plan, execute, assess, control and close the project, being directly related to the Project Planning and Project Assessment and Control Processes of ISO/IEC 12207. Test activities are covered by activities in the Software Implementation Process, in particular Software Architectural and Detailed Design, Software Construction, and Software Integration and Tests, as discussed in Section 2.

For harmonizing the standards' contents, we used ROoST [Souza et al. 2017], the SEON's Software Testing Ontology, as an interlingua. ROoST extends SPO and this enabled us to align structural and content harmonization. We focused on test technical activities, disregarding test management, test environment setting, defects correction, configuration management, and test organizational aspects. Table 2 shows ROoST work units and the standards' mappings to it. The first column presents ROoST concepts. The other columns present the work units of each standard mapped to these concepts, usually with *total* coverage, but some with *partial*, as indicated.

Total coverage here refers to the fact that the standard's work unit is completely covered by the ROoST concept. *Partial* coverage means that the standard's work unit is broader and that there are some portions that are not covered by the ROoST concept. For instance, the Software Construction Process of ISO/IEC 12207 has several activities and tasks that are not related to test, and thus that are not related to any ROoST concept.

Table 2. Content Mappings.

ROoST Concept	ISO 12207 Element	ISO 29110 Element	ISO 29119 Element
Testing Process [Performed Process]	(7.1.5) Software Construction Process {partial} (7.1.6) Software Integration Process {partial} (7.1.7) Software Qualification Testing Process (7.2.5) Software Validation Process {partial}	(SI.3) Software Architectural and Detailed Design {partial} (SI.4) Software Construction {partial} (SI.5) Software Integration and Tests {partial}	(8) Dynamic Test Processes {partial}
Test Case Design [Performed Activity]	(7.1.5.3.1.1) Develop test procedures and data for testing each software unit and database (7.1.6.3.1.4) Develop and document a set of tests, test cases, and test procedures (7.2.5.3.2.1) Prepare selected test requirements, test cases, and test specifications for analyzing test results (7.2.5.3.2.2) Ensure that the artifacts reflect the particular requirements for the specific intended use	(SI.3.5) Establish or update Test Cases and Test Procedures (SI.3.6) Verify and obtain approval of the Test Cases and Test Procedures (SI.4.4) Design or update unit test cases (SI.5.2) Understand Test Cases and Test Procedures (SI.5.3) Update Test Cases and Procedures for integration testing	(8.2) Test Design & Implementation Process (8.2.4.1) Identify Feature Sets (and tasks a,b,c,d,e) (8.2.4.2) Derive Test Conditions (and tasks a,b,c,e) (8.2.4.3) Derive Test Coverage Items (and tasks a,b,c) (8.2.4.4) Derive Test Cases (and tasks a,b,c,e) (8.2.4.5) Assemble Test Sets (and tasks a,b) (8.2.4.6) Derive Test Procedures (and tasks a,b,c,d,f)
Test Execution [Performed Activity]	(7.1.5.3.1.2) Test each software unit and database, and document the test (7.1.6.3.1.2) Integrate the software units and software components and test {partial} (7.1.7.3.1.1) Conduct qualification testing (7.2.5.3.2.3) Conduct the tests previously selected and analyzed (7.2.5.3.2.5) Test the software product as appropriate in selected areas of the target environment	(SI.5.4) Perform Software Tests using Test Cases and Test Procedures for Integration	(8.4) Test Execution Process (8.4.4.1) Execute Test Procedures (and tasks a,b,c) (8.4.4.2) Compare Test Results (and tasks a,b) (8.4.4.3) Record Test Execution (and task a) (8.5) Test Incident Reporting Process (8.5.4.2) Create/Update Incident Report (and tasks a,b)
Test Result Analysis [Performed Activity]	(7.1.5.3.1.5) Evaluate and document the software code and test results (7.1.6.3.1.5) Evaluate and document the integration plan, design, code, tests, test results (7.1.7.3.1.3) Evaluate the design, code, tests, test results, and user documentation (7.2.5.3.2.4) Validate that the software product satisfies its intended use	-	(8.5.4.1) Analyze Test Results (and tasks a,b)

Besides the work units, we also harmonized artifacts (namely: *Test Plan*, *Test Case*, *Test Results*, *Test Case Design Input* and *Code to be Tested*) and stakeholders

(namely: *Test Manager*, *Test Case Designer* and *Tester*). Moreover, taking the harmonized content shown in Table 2, we defined an integrated process, covering the main test-related work units of the harmonized standards. The work products and stakeholders' mappings, as well as the integrated process, are not discussed here due to space limitations.

Defining the content elements and mapping them to create an integrated view is a complex effort requiring semantic analysis and tough decisions. Since ROoST concepts extend SPO concepts, the content mappings were also supported by the structural mappings results, considering only the admissible ones, as shown in Table 1.

Along the contents harmonization some issues were identified. They are related to inconsistencies and divergences observed in the standards that can jeopardize a harmonization effort. Next, we present the main issues detected:

(1) Content Organization: Each standard organize the contents in a different way. While ISO/IEC 29119 presents a number of processes specific for tests, decomposed in detailed activities and tasks, testing content in ISO/IEC 12207 and 29110 is more general and dispersed in different processes and activities devoted to other areas such as software design, construction, integration and validation. Sometimes even small tasks (or artifacts) include other actions (or information) with the testing ones, being hard to extract/analyze the exact testing portion.

(2) Content Presentation: To understand and extract information from text in natural language is not a simple task. The standards are subjective, sometimes providing only information pieces or omitting them. The work units, in general, are well structured; but the work products and stakeholders, when explicitly mentioned, many times are not clearly related to the respective work units. Moreover, the language used varies in some respects, such as writing style, adopted terms and abstraction levels, which makes the comparisons harder.

(3) Granularity: The differences in granularity also affect the contents harmonization. A couple of standards presenting work units with too distinct sizes are harder to map, causing losing in precision. Sometimes several small tasks in a standard have no direct correspondence and need to be all mapped as part of a single bigger concept.

(4) Use of Terms: Inconsistency in the definition or description of some terms can also cause understanding problems and consequently affect the contents harmonization. For instance, ISO/IEC 29110 uses "Document" and "Information Item" for naming artifacts. According to this standard, Information Items correspond to the Test Processes outputs and the Document contains Information Items as shown in the structural model. However, the standard allows dubious interpretation in some parts of the text as to whether the outputs are Documents or Information Items, especially in Parts 2 and 3 of the standard.

As we can notice, in both the structural and contents harmonization we deal with some inconsistencies and divergences between the standards. The identified issues in the contents occur mainly because of differences in the descriptions, as well as the standard subjectivity, making harder to correlate them. As advocated here, such issues can be mitigated by following a semantic referential. Once identified the standards similarities, differences and main issues to be addressed, it is possible to provide an integrated view of the target standards. These results are valuable to understand how the

standards behave in combination, being useful for reducing costs and efforts in organizations aiming at their deployment, and for standardization organizations to provide more aligned standards. Finally, the adopted approach favors the definition of an integrated process, unifying the selected aspects of each standard.

5. Related Works

Several works have been conducted for harmonizing software engineering standards. Jeners et al. (2013) propose an approach based on metamodels and comparison techniques for mapping and integrating standards. They harmonize general software process (such as CMMI-DEV) and governance (such as COBIT) standards. Pardo et al. (2013) propose a harmonization framework with a detailed process, making use of two ontologies, for managing and executing harmonization initiatives for multimodel environments. They present some applications, mostly using IT Governance standards for the Bank sector. Henderson-Sellers et al. (2014), as part of an ISO harmonization initiative, propose an ontological framework as a reference infrastructure for harmonizing the ISO SC7 standards. Although these works are general and applicable for diverse software engineering related standards, we have not found studies specifically discussing the issues around the testing process.

Considering our approach, we can make comparisons regarding aspects coming from the findings of a systematic literature mapping we have conducted on Software Engineering standards harmonization by means of common models. Like we do, many works consider harmonization in two perspectives, structural (usually already embedded in the approach) and content, for effectively analyzing the standards knowledge (e.g., [Ferchichi et al. 2008] [Jeners et al. 2013] [Pardo et al. 2013]). Regarding the techniques, the most frequently applied are homogenization, mapping and integration. Few works combine the three techniques, starting from the standards information extraction, organizing and mapping it, and producing an integrated artifact (e.g., [Ferchichi et al. 2008] [Pardo et al. 2015]). A trend perceived in the analyzed works is the increasing adoption of semantic referential for supporting the harmonization efforts (e.g., [Henderson-Sellers et al. 2014] [Pardo et al. 2015]). Although those works use metamodels and ontologies with some variation, the purposes converge to the same: relying on a consistent representation of the domain semantics. In this sense, we are working on a more advanced solution, by using SEON, an ontology network fully grounded in UFO (a foundational ontology) and with a core ontology on software processes and diverse domain specific ontologies, able to address standards regarding several domains and description levels. Finally, the majority of the harmonization works focus on supporting software organizations aiming at deploying multiple standards (e.g., [Pardo et al. 2015] [Mejia et al. 2016]). Our focus is, besides helping the standards users, to make more evident the standards similarities and misalignments, providing useful material also for supporting standards developers in future standards reviews.

6. Conclusions

This paper presented an initiative harmonizing software testing practices from three ISO standards. It was supported by ontologies and conducted in two steps, focusing on structural and content aspects. The standards similarities and differences were analyzed, and the main result is a mapping between the standards contents, using of a software testing ontology as *interlingua*. Along the harmonization efforts, diverse issues arose

regarding standards divergences and inconsistencies. The results produced and the identified issues are useful for SPI initiatives in multimodel environments and also for future improvements in the standards.

The harmonization of multiple standards involves a large amount of information from the standards, which was captured, analyzed and mapped. Although we present results covering most of the investigated aspects, some were omitted (due to space limitations) and part of the test scope was disregarded in this study. For example, concerning structural aspects we have not considered notions referring to very different things, such as *outcome*. However, this is an important element, mainly in certification standards, and thus we intend to address this notion in future works. Other example is *task*, which can represent many different things, but we considered only its work unit facet. Considering the contents, we have limited the scope as described, but we intend to extend it in future works.

The use of a fragment of SPO for harmonizing the standards' structures, and ROoST for harmonizing standards' contents showed valuable benefits in considering semantics. Thus, as an ongoing work, we are developing an ontology-based approach for standards harmonization, improving and systematizing the approach followed here.

Acknowledgements

This research is funded by the Brazilian Research Funding Agency CNPq (Process 461777/2014-2) and FAPES (Process 69382549/2014).

References

Biffl, S., Winkler, D., Höhn, R. and Wetzel, H. (2006). Software Process Improvement in Europe: Potential of the New V-model XT and Research Issues. *Software Process: Improvement and Practice*, 11(3), pp.229-238.

Bourque, P. and Fairley, R. E. (2014). *Guide to the Software Engineering Body of Knowledge, SWEBOK, Version 3.0*. IEEE Computer Society Press.

Falbo, R.A. and Bertollo, G. (2009). A Software Process Ontology as a Common Vocabulary about Software Processes. *International Journal of Business Process Integration and Management*, 4(4), pp.239-250.

Ferchichi, A., Bigand, M. and Lefebvre, H. (2008). An Ontology for Quality Standards Integration in Software Collaborative Projects. In: *Proceedings of the First International Workshop on Model Driven Interoperability for Sustainable Information Systems (MDISIS'08)*, pages 17–30, Montpellier, France.

García, F., Bertoa, M., Calero C., Vallecillo, A., Ruíz, F., Piattini, M. and Genero, M. (2006). Towards a Consistent Terminology for Software Measurement. *Information and Software Technology*, 48 (8), pp.631-644.

Guizzardi, G., Falbo, R.A., Guizzardi, R.S.S. (2008). Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The Case of the ODE Software Process Ontology. In: *Proceedings of the XI Iberoamerican Workshop on Requirements Engineering and Software Environments*, pp. 244-251. Recife, Brazil.

Henderson-Sellers, B., Gonzalez-Perez, C., Mcbride, T. and Low, G. (2014). An Ontology for ISO Software Engineering Standards: 1) Creating the Infrastructure. *Computer Standards & Interfaces*, 36(3), pp. 563-576.

IEEE (2004). IEEE Standard for Software Verification and Validation, Std 1012, New York, USA.

ISO/IEC (2007). ISO/IEC 24744 - Software Engineering – Metamodel for Development Methodologies.

ISO/IEC (2008). ISO/IEC 12207 - Systems and Software Engineering – Software Life Cycle Processes.

ISO/IEC (2011). ISO/IEC TR 29110 - Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs).

ISO/IEC (2013). ISO/IEC 29119 - Software and Systems Engineering - Software Testing (Parts 1, 2, 3 and 4).

Jeners, S., Lichter, H. and Rosenkranz, C.G. (2013). Efficient Adoption and Assessment of Multiple Process Improvement Reference Models. *e-Informatica Software Engineering Journal*, 7(1).

Mathur, A. P. (2012). *Foundations of Software Testing*. 5 ed. India: Dorling Kindersley (India), Pearson Education in South Asia.

Mejia, J., Muñoz, E. and Muñoz, M. (2016). Reinforcing the Applicability of Multi-Model Environments for Software Process Improvement Using Knowledge Management. *Science of Computer Programming*, 121, pp. 3-15.

Pardo, C., Pino, F.J., Garcia, F., Baldassarre, M.T. and Piattini, M. (2013). From Chaos to the Systematic Harmonization of Multiple Reference Models: A Harmonization Framework Applied in Two Case Studies. *Journal of Systems and Software*, 86(1), pp.125-143.

Pardo. C., García, F., Piattini, M., Pino, F.J. and Baldassarre, M.T. (2015). A 360-degree Process Improvement Approach based on Multiple Models. *Revista Facultad de Ingeniería Universidad de Antioquia*, (77), pp.95-104.

Paulk, M.C. (1993). Comparing ISO 9001 and the Capability Maturity Model for Software. *Software Quality Journal*, 2(4), pp.245-256.

Ruy, F.B., Falbo, R.A., Barcellos, M.P., Costa, S.D. and Guizzardi, G. (2016). SEON: A Software Engineering Ontology Network. In Proceedings of 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW'16), Bologna, Italy, pp.527-542.

SEI (2010). CMMI for Development, Version 1.3. CMU/SEI-2010-TR-033. Software Engineering Institute, Carnegie Mellon University.

SOFTEX (2016). *Brazilian Software Process Improvement (MPS.BR) - General Guide: 2016*, Brazil.

Souza, É.F.D., Falbo, R.D.A. and Vijaykumar, N.L., (2017). ROoST: Reference Ontology on Software Testing. *Applied Ontology*, 12, pp. 59-90.

Studer, R., Benjamins, R. and Fensel, D. (1998). Knowledge Engineering: Principles and Methods. *Data & Knowledge Engineering*, 25(1–2), pp. 161–198.

TMMi Foundation (2012). Test Maturity Model integration (TMMi) 1.0, Ireland.

Yoo, C., Yoon, J., Lee, B., Lee, C., Lee, J., Hyun, S. and Wu, C. (2006). A Unified Model for the Implementation of both ISO 9001:2000 and CMMI by ISO-certified Organizations. *Journal of Systems and Software* 79 (7), pp. 954–961.