

# My Path experiencing *Falbo's Approach*

Monalessa Perini Barcellos

Ontology and Conceptual Modeling Research Group (NEMO)  
Computer Science Department – Federal University of Espírito Santo, Vitória, Brazil  
[monalessa@inf.ufes.br](mailto:monalessa@inf.ufes.br)

**Abstract.** When I was an undergraduate student, I heard a lot about a certain professor, referred to my colleagues as the *best professor* they had, and I was sorry I did not have classes with him because he had taken a leave to obtain his doctor degree. Years later, I met that professor. I not only met him, but I had the honor of being his first doctorate student. At that moment, I found out that he was much more than the *best professor*. Being advised by him I learned so much as a person and as a student. After finishing my doctorate, I received the immeasurable gift of having this professor as my greatest partner at work and, above all, my great friend. Again, I confirmed that *best professor* was too little to him. What *Ricardo de Almeida Falbo* does makes him much more than that! Working with him is a continuous learning. He makes people better. I can say that from my own experience. More than ten years working with him and being friends with him made me a better person and a better professional. His generosity, competence, patience, excellence, zeal, wisdom, friendship and values make him much more than the *best professor*. He has that “extra” that turns an ordinary person into an *extraordinary person*. To me he always will be a *role model*, my *great friend*, a *mentor*, and a *great master*. This paper is a tribute to him, on the occasion of his formal retirement. Here, I summarize some works we have developed together along the last years.

## 1 Introduction

Writing this paper was such a pleasure! I spent hours recollecting many years of meetings, discussions, emails, messages, papers, conceptual models and specifications, and all those versions (there were so many! v0.1, v0.2... v0.25...!) of artifacts we produced until finally achieving the “v1.0”, as a result we found good. Our meetings and discussions have always been productive and enjoyable. We used to have some fun even when the subject was hard. In fact, seeing the bright side of things is a hallmark of Falbo (and it is contagious!). After a while recollecting moments shared with Falbo along the path we went through together in the last years, I selected some results to report here. This paper is a summary of some works resulting from our partnership in the last years. Many other people are also involved in these works: students, colleagues, partners.

Years ago, a friend of ours found a paper that mentioned what was called “*Falbo’s Approach*”. Since then, in our research group, we have often used this term as a kind of joke, to play with Falbo. However, there is a “*Falbo’s Approach*” indeed. It is not only about his academical proposals. It is the way Falbo do things. It is the way he works, thinks, acts, values his friends, shares knowledge, sees life, chooses what does and what does not matter. Working with Falbo comes with the privilege of experiencing *Falbo’s Approach*.

The works reported in this paper have a subject in common: *ontologies*. An ontology is a formal representation of a common conceptualization of a universe of discourse [1]. It is a useful instrument for reducing conceptual ambiguities and inconsistencies, and for making knowledge structures clearer. Ontologies can be classified according to their generality level into: *foundational ontologies*, which describe very general concepts, such as object, event, etc.; *domain ontologies*, which describe the conceptualization related to a domain (e.g., medicine, law); *task ontologies*, which describe the conceptualization related to a task or process (such as diagnosis and sale); and *application ontologies* that describe concepts dependent on a particular domain and task [1]. *Core ontologies* are between foundational and domain ontologies, providing a precise definition of structural knowledge in a specific field that spans across different application domains in this field [2].

Another important distinction differentiates ontologies as conceptual models, called *reference ontologies*, from ontologies as computational artifacts, called *operational ontologies*. A reference ontology is constructed with the purpose of making the best possible description of the domain in reality, representing a model of consensus within a community, regardless of its computational properties. Operational ontologies, in turn, are designed with the focus on guaranteeing desirable computational properties [3].

The works I report here were developed in the *Ontology and Conceptual Modeling Research Group* (NEMO) and used UFO (*The Unified Foundational Ontology*) [4, 5, 6] as foundational ontology. UFO is based on a number of theories from Formal Ontology, Philosophical Logics, Philosophy of Language, Linguistics and Cognitive Psychology. It is divided in three parts: an ontology of endurants (objects) [4], an ontology of perdurants (events) [5], and an ontology of social entities [6].

In the last years, I have worked with Falbo in two main areas: *Software Engineering* and *Ontology Engineering*. In the former, we have investigated how to apply ontologies to support software-related processes. In the latter, we have studied how to support ontology development, mainly by facilitating reuse. More specifically, in the Software Engineering context, we have developed several ontologies and applied them to support systems integration, semantic documentation and standards harmonization. In the Ontology Engineering context, we have proposed to develop ontologies as ontology networks and organize ontology patterns in Ontology Pattern Languages [7] in order to favor reuse.

This paper does not intend to present the works in detail. Contrarywise, it aims at giving a general view of some research topics that have been influenced by Falbo and research results that have been produced with his participation. Detailed information about the works mentioned here can be found in the respective references.

The paper is organized as follows. Sections 2, 3 and 4 regard the Software Engineering context, addressing, respectively, the Software Measurement Ontology, my first work with Falbo; Semantic Interoperability; and SEON, a Software Engineering Ontology Network. Section 5 regards Ontology Engineering, focusing on Ontology Pattern Languages. Last, in Section 6, I present final considerations and personal notes.

## 2 Software Measurement Ontology

Everything started in 2008. I was working with software measurement and statistical process control in my doctorate research project, advised by professor Ana Regina Cavalcanti da Rocha, at COPPE/UFRJ, and I was looking for something innovative to improve my proposal. At that time, statistical process control was recent in software organizations and there were still many doubts about it. Discussing with Falbo about my research, he introduced me to the *Ontologies* subject and provided me the first materials I read about the topic, which included a paper by Guarino [1], and some papers and a thesis about UFO [4]. He talked to me about well-founded ontologies, the benefits of using them, and also about some applications he had done in the Software Engineering domain. After some discussion, I noticed that I could propose a reference ontology to provide a well-founded conceptualization about software measurement and statistical process control and explore it to help organizations in these practices. That would be great! Moreover, I could improve my research and start a new partnership.

Falbo accepted the invitation to be my co-adviser (Ana Regina Rocha and I are very grateful for that) and we worked for months developing the *Reference Software Measurement Ontology* (RSMO) [8, 9, 10, 11, 12], which is grounded in UFO [4, 5, 6]. I lost count of the number of versions were produced! When we thought we have finished, a new question came up and we got back to the ontology. How much I learned in this process!

After working with Falbo as his advisee, I had the honor to work with him as work partners. Once we developed and used RSMO, we noticed that it would also be important to represent knowledge about the software measurement process, highlighting behavioral aspects that are not explicit in domain ontologies. Therefore, we built the *Task Software Measurement Ontology* (TSMO) [13]. RSMO and TSMO were applied in practice to support organizations to understand software measurement and statistical process control and also to help better understanding and thus harmonizing different measurement-related standards [13]. These applications were particularly relevant to software organizations that were interested in implementing practices from CMMI (Capability Maturity Model Integration) [14] levels 4 and 5, and MR MPS (Reference Model for Brazilian Software Process Improvement) [15] levels B and A. Hence, these works contributed to researchers and practitioners.

### 3 Semantic Interoperability

Over the years, our partnership was extended and involved many other people. We have worked in several research projects and advised together several students. One of the main problems we have explored in these works refers to *semantic interoperability*.

Nowadays, information is one of the most important assets for people and organizations. However, most often, information is spread in different and heterogeneous sources (e.g., different documents, standards, web pages, models, information systems etc.) and it may not be easy to combine them and obtain useful information that properly supports decision making.

Obstacles for interoperability arise mainly from the fact that the information sources are usually created independently and do not share the same semantics in the used terminology. Different terms can be used to represent the same concept, while the same term can be used to represent different concepts. Without an explicit definition of the involved terms, it is difficult to know the correspondence between concepts from different information sources. However, sharing a terminology is not enough. The entities must share semantics, i.e., the meaning of the used terminologies.

Semantic interoperability, in general, refers to the ability to exchange information based on meaning. In several contexts, semantic conflicts arise because the things being integrated (systems, data, models and so on) do not share a common conceptualization. Thus, to solve semantic conflicts, ontologies can be used as an interlingua to map concepts from different sources.

In the last years, we have investigated semantic interoperability in three main contexts: *systems integration*, *semantic documentation* and *standards harmonization*.

#### 3.1 Systems Integration

Systems integration is one of the biggest challenges faced by organizations to obtain consolidated information from several sources. Taking that into account, Calhau and Falbo developed OBA-SI (*Ontology-Based Approach for Semantic Integration*) [16], which proposes the use of ontologies to assign semantics and support mappings to integrate systems. OBA-SI defines an integration process similar to the software development process, comprising six phases. *Integration Requirements Elicitation* consists of establishing the integration requirements, identifying the business activities that will be supported by the integrated solution, and the systems to be integrated to support those activities. *Integration Analysis* is the phase in which the 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. It involves using ontologies as reference models to map the systems models and produce the integration model to be used in the solution. Last, there are the *Design*, *Implementation*, *Tests*, and *Deployment* phases, when the conceptual solution previously modelled is turned into an operational solution [16].

After some experiences applying OBA-SI [16] in practice, we identified some improvement opportunities. First, when we tried to apply OBA-SI to integrate systems to support the software measurement process, we noticed that this domain has some particularities that demand specific activities and ontologies when using OBA-SI.

Therefore, we decided to specialize OBA-SI to the software measurement domain. Second, we noticed that although OBA-SI satisfactorily addresses data integration, service and process integration are not so clear. Hence, we carried out investigations and detailed integration at these layers.

### **Integrating Systems to Support Software Measurement**

Typically, software organizations use different systems to support different processes. For example, schedule and budget systems are used to support project management; modeling tools are used to support requirements engineering; and development environments and version control systems are used to support coding and source code management. Although these systems 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, number of lines of code, test failure rate, etc.). In order to properly support the software measurement process, these systems must be integrated. However, this is not an easy task. The heterogeneity between systems to be integrated is the major difficulty. In general, each system runs independently and implements its own data and behavioral models, which are not shared between different systems, leading to several conflicts [17].

Therefore, we developed the *Ontology-Based Approach for Measurement Systems Integration* (OBA-MSI) [18], a systematic approach that uses the Reference Software Measurement Ontology (RSMO) [8, 9, 10, 11, 12] and the Software Measurement Task Ontology (SMTO) [13] to guide systems integration to support the software measurement process. OBA-MSI specializes OBA-SI [16]. OBA-SI can be applied to carry semantic integration of systems in any domain. However, systems integration in the software measurement domain has some peculiarities that are not properly addressed by OBA-SI. For instance, OBA-SI is more suitable for integrating applications that support the same process. However, systems integration to support the software measurement process typically involves applications designed to support different processes that must be integrated to the software measurement process. Besides, in OBA-SI, integration requirements elicitation is superficially addressed. In OBA-MSI, integration requirements elicitation is detailed following a goal-based approach. Figure 1 shows the first phases of the OBA-MSI process<sup>1</sup>. The last phases are the same than OBA-SI. Figure 2 details the first OBA-MSI phase. The complete description of OBA-MSI is available at [18].

By specializing OBA-SI to the software measurement domain, three main contributions were given: (i) OBA-SI general steps were detailed and specialized to the case of software measurement, resulting in more palatable steps to be followed by users; (ii) we defined an integration approach more suitable for integrating systems developed with the aim of supporting very different software processes, but that produce input for the software measurement process; and (iii) an ontology framework made up of RSMO and SMTO was provided. One of the most effort-demanding step in OBA-SI is related to select (or develop) and integrate ontologies to be used in the integration initiative. In OBA-MSI, the ontologies to be used are provided and the effort is reduced to understand these ontologies.

---

<sup>1</sup> In this paper, the words *tool* and *system* are used as synonymous.

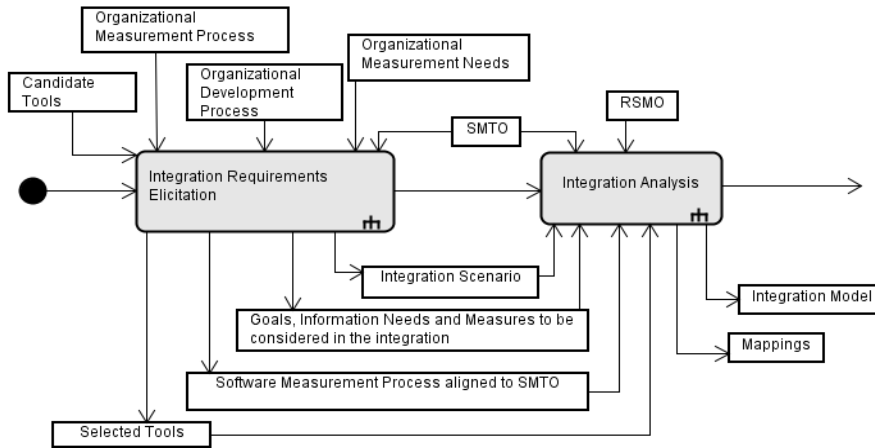


Fig. 1. First phases of OBA-MSI process.

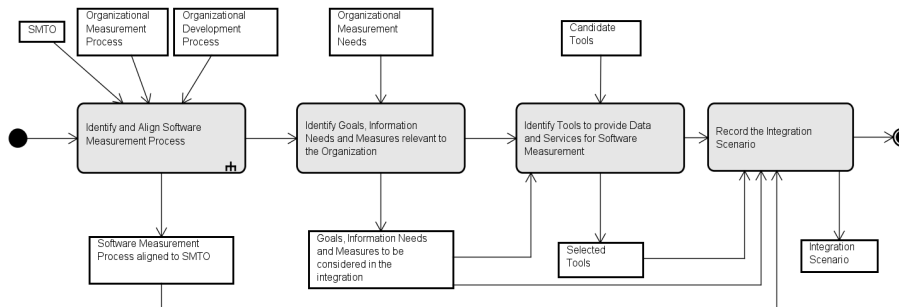


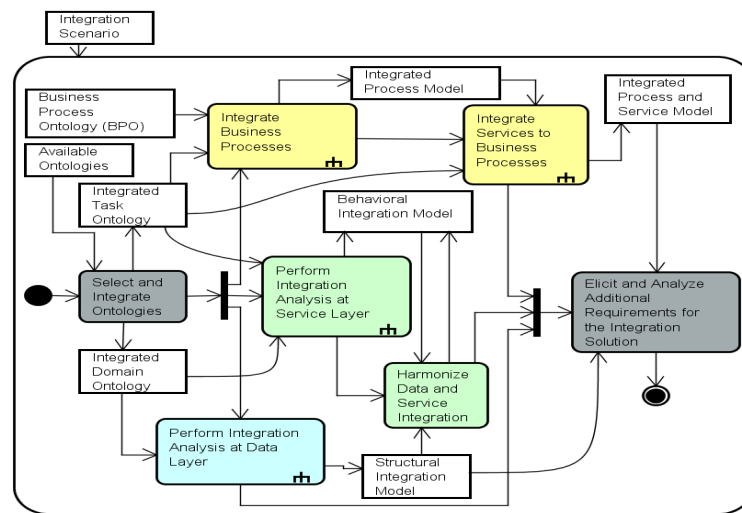
Fig. 2. OBA-MSI - Integration Requirements Elicitation phase.

### Integrating Systems at Data, Service and Process Layers

Systems integration can be performed at three layers [17]: data, message, and process. *Data integration* deals with moving or federating data between multiple data stores. It assumes bypassing the application logic and manipulating data directly in the database. *Message (or service) integration* addresses messages exchange between the integrated applications. *Process integration*, in turn, views organizations as a set of interrelated processes and it is responsible for handling message flows, implementing rules and defining the overall process execution.

Our experiences using the first version of OBA-SI [16] showed us that data integration was well addressed, but service and process integration should be improved. When we developed OBA-MSI [18], we explored the Task Software Measurement Ontology (TSMO) [13] to address service and process integration. From these and other experiences, we decided to evolve OBA-SI [16] by improving integration at service and

process layers using task ontologies to assign semantics at these layers. Moreover, we argue that integration at process layer should consider a common understanding about business processes. Therefore, we proposed a *Business Process Ontology* (BPO) [19], a well-founded ontology grounded in UFO [4, 5, 6], which provides a common conceptualization about business process, to be used to support process integration. Figure 3 shows the overview of the Integration Analysis phase of OBA-SI after the improvements we made. A detailed description of this phase and of the current version of OBA-SI is made in [19].



**Fig. 3.** OBA-SI - Integration Analysis phase addressing all layers [19].

OBA-SI [19] uses domain and task ontologies in different activities. Domain ontologies are used at both data and service layers. At data layer, domain ontologies' concepts, relations and properties are used to assign semantics to classes, associations and attributes of the systems' data conceptual models. For addressing integration at service and process layers, OBA-SI uses task ontologies. At service layer, domain ontology concepts are used to assign semantics to inputs and outputs of the functionalities/services provided by the systems being integrated. Task ontologies, in turn, are used to assign semantics to the systems' functionalities/services and to the business processes' activities. It is possible that more than one domain or task ontology are necessary to cover the domains or processes involved in the integration initiative. In this case, the ontologies must be integrated in a way that results in a single domain ontology and a single task ontology to be used in the integration initiative. Moreover, when both domain and task ontologies are needed, they must be integrated, giving rise to an application ontology, involving both domain and task perspectives.

With the improvements made in OBA-SI, in its current version [19] the integration process is clearer, guiding users on the activities to be performed and allowing users to understand the relations between the integration layers and address only the layers of

interest in a particular system integration initiative. Moreover, semantics assignment at service and process layers was improved, since task ontologies are more suitable for dealing with behavioral aspects than domain ontologies.

### 3.2 Semantic Documentation

In the context of software projects, documents hold a considerable amount of information that is mainly interpreted by humans [20]. One disadvantage of using documents is the difficulty of obtaining consolidated information from them, especially when information is spread in several documents. Accessing, recovering and managing information recorded in documents usually depend on human intervention and can be laborious and error-prone. Besides, gathering relevant information from different documents can be so wearing that people may tend not to do so [21].

Semantic Web deals with a similar issue by providing a way that both humans and computers can interpret the content of web pages. To reach this goal, web pages are annotated with metadata that describe fragments of the page content so that they become available for computer interpretation [22]. Since ontologies represent a conceptualization about the domain of interest and establish a common vocabulary to be shared, they are of great value to describe metadata, providing a rich formal semantic structure for their interpretation. Therefore, ontologies are often used as a basis for semantic annotation [23].

Semantic Web principles can also be applied to documents (e.g., documents produced in desktop text editors, electronic spreadsheets), giving rise to *Semantic Documentation*. Semantic Documentation combines documents and ontologies in the same way that Semantic Web, i.e., ontology-based metadata are created and then attached to the document content, resulting in a semantic document [24]. By doing that, syntactic information resources are turned into semantic information resources, which can be interpreted by computers. Once information recorded in the documents is automatically retrieved by computers, it is possible to develop tools to provide consolidated and integrated information for end users, decreasing the human effort necessary to obtain such information. Moreover, semantic annotation approach allows relating annotated contents and using the relationships to extract information from several documents, providing a general view that probably could not be gotten without the annotations [21].

Figure 4 illustrates the semantic annotation process for generating a semantic document. The person responsible for adding metadata to documents uses a supporting tool to semantically enrich the document. For each annotation, the tool creates semantic metadata relating a fragment of the document (referring to general information or document content) to an element of the ontology. Usually, metadata are kept in the semantic document. Once documents are annotated, it is possible to extract knowledge and link contents from different documents according to the shared ontology. By integrating content extracted from several documents, it is possible to achieve a more holistic view of the available knowledge [21].

Aiming at supporting semantic documentation, Falbo and Arantes [21] proposed an environment to manage desktop semantic documents, allowing semantic annotation of document templates. Thus, documents produced by using the annotated templates are



automatically annotated, easing end users work, since they can use the annotated templates to create semantic documents without concern with the annotations to be made. Figure 5 shows an overview of the infrastructure for managing semantic documents proposed by Falbo and Arantes [21].

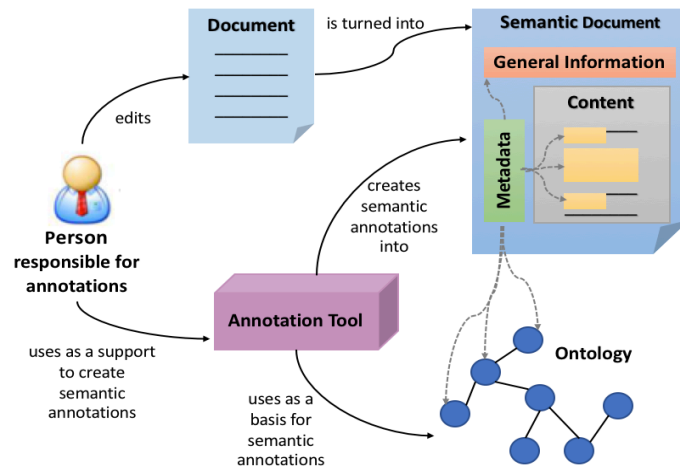


Fig. 4. OBA-SI Creating semantic documents [25].

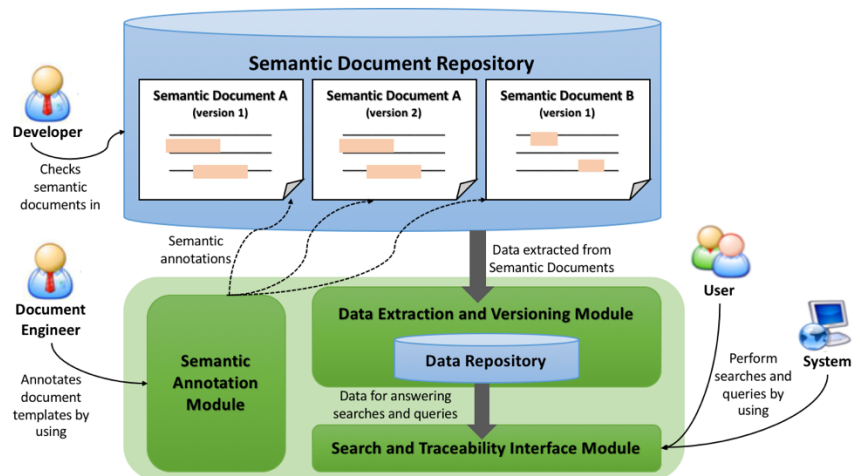


Fig. 5. Overview of the Infrastructure for Managing Semantic Documents [21].

Considering some industrial reports and experiences we had in some organizations, we noticed that many organizations adopt documents and spreadsheets to record information about software projects and to manage them. For example, during a software project, information regarding planning, execution, progress monitoring and control many times is recorded in text documents and spreadsheets (e.g., project plan and status

reports). Due to the importance of these documents for a project, project members should share them in a common environment and be able to access their content in an easy and efficient way [26]. Once documents are annotated, it is possible to build repositories containing semantic documents produced in the projects and develop semantic documentation systems able to retrieve consolidated information from them. Besides, semantic documentation helps store and retrieve knowledge acquired during a project and reuse it in other projects. Thus, we decided to explore the use of semantic documentation in software project management and provide an environment to support organizations in this context.

To provide the domain conceptualization, we developed the *Software Project Management Ontology* (SPMO) [25], grounded in UFO [4, 5, 6]. We used SPMO and extended the infrastructure proposed by Falbo and Arantes [21] to develop the *Infrastructure for Managing Semantic Documents for Software Project Management* (IMSD-SPM) [25], a semantic documentation infrastructure with specific features to support project management aspects. From data recorded in semantic documents and spreadsheets related to scope, time and cost produced along projects, features provide to managers: (i) a consolidated view of project planning regarding scope, time and cost; (ii) dependency matrices, showing dependency relations among project elements; (iii) consolidated information about project execution, pointing out the differences between planning and execution values; (iv) project performance indicators; (v) estimates for project conclusion; (vi) a global view of the performance of several projects, allowing for comparisons among them; and (vii) non-conformities detected in the semantic documents and spreadsheets content.

IMSD-SPM helps software organizations to get consolidated information from data recorded in different documents and spreadsheets. This contributes to well-informed decision making. It is important to point out that problems to retrieve information from documents and spreadsheets could be also addressed by approaches such as corporative architectures and information systems. However, although these approaches are capable of dealing with information recording and extraction, they imply in modifying the way organizations and people perform activities, since they replace documents by information systems. For organizations that use documents and spreadsheets (and prefer to keep this practice), using semantic documentation-based solutions, such as IMSD-SPM, can be an advantageous approach, because it accesses information recorded in the documents, allowing organizations and people to keep the way they perform activities.

### **3.3 Standards Harmonization**

In the Software Engineering domain there are several standards produced by organizations such as ISO (International Standardization Organization), IEC (International Electrotechnical Commission), IEEE (Institute of Electrical and Electronics Engineers), SEI (Software Engineering Institute) and SOFTEX (Association for Promotion of Brazilian Software Excellence). Standards are produced in different moments, by different groups and with different purposes. Often, organizations need to combine different standards to achieve a certain goal. However, the standards do not share a

common conceptualization, and integrating them is not trivial. Thus, standards harmonization is essentially a semantic interoperability problem, as we can observe very interrelated information described in different and sometimes conflicting ways by distinct sources (standards).

We have worked on standards harmonization for several years. For example, in [13] we used TSMO as reference model to analyze and harmonize software measurement standards. After many experiences analyzing different standards and defining software processes that should be in conformance with several of them, we noticed that a systematic approach would be helpful. In 2015, Falbo was advising a doctorate student and the research topic was related to standards harmonization. At that time, we had already carried out an ontological analysis of de ISO/IEC 24744 [27] and participated in an ISO work group together. Thus, Falbo invited me to work as co-adviser of his student. Until then, we had worked together advising master students, but that was our first partnership as co-advisers of a doctorate student. The results were very promising. One of them, *Harmony* [28], is a systematic approach for conducting standards harmonization initiatives that uses the *Software Engineering Ontology Network* [29] (described below) for representing the standards and applying harmonization techniques. *Harmony* was developed considering knowledge from the literature and harmonization initiatives we experienced in the last years (e.g., [30]). It is based on three main actions: modeling, which refers to representing the standards as models (in opposition to text) for better dealing with them; mapping, which regards mapping the standards models to ontologies serving as a semantic referential; and integration, which concerns building a unified view of the standards by extending the domain view with the necessary standard's specific elements.

## 4 Software Engineering Ontology Network

Software Engineering (SE) is a wide domain, where ontologies are useful instruments for dealing with knowledge management related problems. When SE ontologies are built and used in isolation, some problems remain, in particular those related to knowledge integration. Along the years, we have produced several ontologies related to SE subdomains, such as software measurement [8, 9, 10, 11, 12], software process [31], requirements [32], configuration management [33], software project management [25], test [34] and others. We experienced some problems, such as the same concept appearing with different meanings in different ontologies and the same term being used to designate different concepts in different ontologies, among others.

In large and complex domains, as is the case of SE, if we try to represent the whole domain as a single ontology, we will achieve a large and monolithic ontology that is hard to manipulate, use, and maintain [35]. On the other hand, representing each subdomain separately would be too costly, fragmented, and again hard to handle.

D'Aquin and Gangemi [36] point out a set of characteristics that are presented in "beautiful ontologies", from which the following ones can be detached: having a good domain coverage; being modular or embedded in a modular framework; being formally

rigorous; capturing also non-taxonomic relations; and reusing foundational ontologies. Most of the existing SE ontologies do not exhibit such characteristics.

Therefore, we started to investigate how to develop and organize ontologies in an architecture that enables integrating existing ontologies and adding new ontologies, keeping consistence between them. Considering the characteristics cited above, we proposed to organize SE ontologies in an ontology network that supports creating, integrating and evolving its ontologies. An *Ontology Network* (ON) is a collection of ontologies related together through a variety of relationships, such as alignment, modularization, and dependency. A *networked ontology*, in turn, is an ontology included in such a network, sharing concepts and relations with other ontologies [35].

To truly enjoy the benefits of keeping the ontologies in a network, we need to take advantage of the existing resources available in the ON for gradually improving and extending it. Thus, an ON should have a robust base equipped with mechanisms to help its evolution. We proposed to organize the ON in layers. Briefly, in the background, we need a *foundational ontology* to provide the general ground knowledge for classifying concepts and relations in the ON. In the center of the ON, *core ontologies* should be used to represent the general domain knowledge, being the basis for the subdomain networked ontologies. Finally, going to the borders, (sub) *domain ontologies* appear, describing the more specific knowledge.

After defining the architecture, we have developed SEON [29]. SEON provides a well-grounded network of SE reference ontologies, and mechanisms to derive and incorporate new integrated subdomain ontologies into the network. In SEON, UFO [4, 5, 6] lies in the foundation layer, providing the common grounding for all the networked ontologies. At the core layer, there is the *Software Process Ontology* (SPO) [31], which provides a common conceptualization about software process, embracing the following aspects of the software process domain: standard, project and performed processes and their activities, artifacts handled, resources used and procedures adopted by activities, team membership, and stakeholders allocation and participation in activities. Two external core ontologies are also integrated to SEON: the *Enterprise Ontology* (EO) [37], which addresses general concepts related to organizations; and the *Core Ontology for Measurement* (COM) [38], which deals with measurement-related general concepts. Finally, at the domain level, there are several ontologies addressing aspects related to SE subdomains.

Figure 6 shows the current status of SEON. Each circle represents an ontology. The circle size varies according to the ontology size (in terms of number of concepts, represented inside the circles in parenthesis). Arrowed lines denote dependencies between networked ontologies, and line thickness represents the coupling level between them (in terms of number of relationships between concepts in different ontologies). The blue circle represents a core ontology. Green circles represent the domain ontologies. The upper-left area comprising four requirements domain ontologies represents the ReqON subnetwork. EO and COM are not represented in the figure because they are external to SEON.

It is important to notice that, even adopting a layered architecture, SEON is a network and each new added node contributes for the whole network. When a new ontology is added, it should reuse existing elements (from a higher or the same layer). Other

ontologies, in turn, may be adapted to keep consistency and share the same semantics along the whole network. Even the core ontologies can evolve to adapt or incorporate new concepts or relations discovered when domain ontologies are created or integrated.

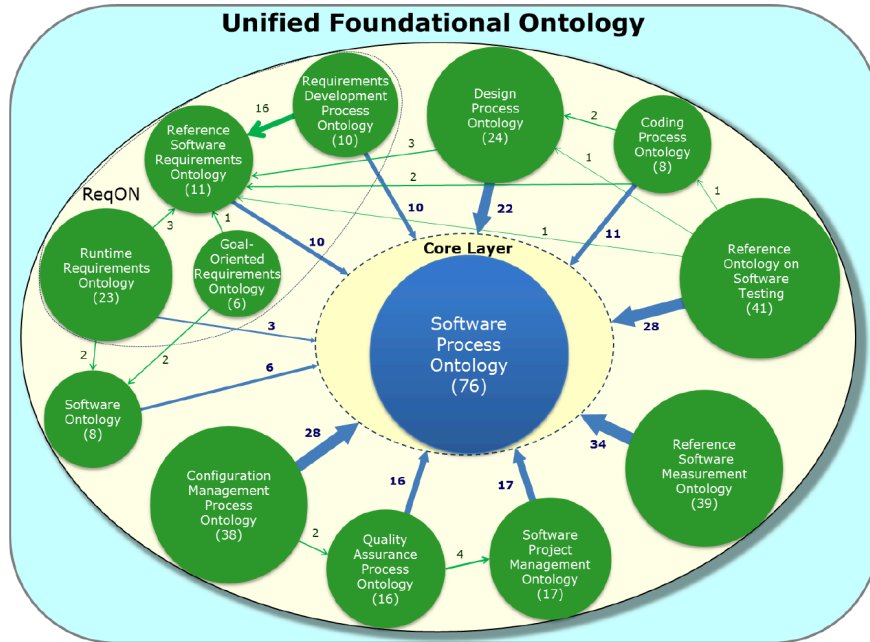


Fig. 6. SEON: The network view [28].

Being an ontology network, SEON is like a living organism and is constantly evolving. It requires a continuous and long-term effort with ontologies being added and integrated incrementally. Therefore, we have been continuously working on SEON. SEON specification is available at [nemo.inf.ufes.br/projects/seon](http://nemo.inf.ufes.br/projects/seon), where a machine processable lightweight version implemented in OWL is also available. Some experiences and envisioned applications of SEON are discussed in [29].

## 5 Ontology Pattern Languages

The works reported in the previous sections are concerned mainly to ontologies applied to Software Engineering. The success of using ontologies to solve knowledge-related or semantic interoperability problems is related to the quality of the used ontologies. The quality of an ontology, in turn, is strongly related to the quality of the languages, methods and tools used to develop it. Thus, it is important to advance on the theoretical and practical support for ontology engineering. In this context, Falbo has contributed to the ontology community by defining methods such as SABiO (*Systematic Approach for Building Ontologies*) [39], which supports ontology development by providing a set of process and activities to be followed to produce reference and operational ontologies.

Moreover, the architecture and framework defined to SEON can be used to define ontology networks related to other areas.

One of the processes prescribed by SABiO is Reuse, which has been pointed out as a promising approach for ontology engineering, helping speed up the development process and improve the quality of the resulting ontologies [40]. In this context, *ontology patterns* are an emerging approach that favors the reuse of encoded experiences and good practices. They are modeling solutions to solve recurrent ontology development problems [41]. We noticed that ontology patterns are often made available in catalogs. This does not favor reuse, since the sense of connection between patterns is lost. From our Software Engineering background, we introduced the *Ontology Pattern Language* (OPL) notion [7]. An OPL is a network of interconnected domain-related ontology patterns that provides holistic support for solving ontology development problems for a specific domain. An OPL contains a set of interconnected patterns, plus a modeling workflow guiding on how to use and combine them in a specific order, and suggesting patterns for solving some modeling problems in that domain [7].

It is important to highlight that we borrowed the term “pattern language” from Software Engineering, where patterns have been studied and applied for a long time. A pattern language, in a Software Engineering view, is a network of interrelated patterns that defines a process for systematically solving coarse-grained software development problems [42, 43]. Thus, we are not actually talking about a “language” properly speaking.

An OPL should indicate explicitly which referenced patterns address mandatory aspects and which ones address optional aspects. To ensure a stable and sound pattern application, referenced patterns should be presented in the suggested application order. Without this explicit procedural guidance, a representation that fits the basic network of the patterns might not provide a suitable process that helps to ensure a sufficiently complete and well-formed ontology.

OPLs are structured to support and encourage the application of one pattern at a time, in the order defined by the pattern sequences that result from the chosen paths through the language. This guideline ensures that the main property of piecemeal growth is preserved: the ‘whole’ always precedes its ‘parts’. A pattern language is of little use if its audience loses the big picture. Conversely, the essential information of each individual pattern within the language must still be preserved [43].

In summary, an OPL should give concrete and thoughtful guidance for developing ontologies in a given domain, addressing at least the following issues: (i) What are the key problems to solve in the domain of interest? (ii) In what order should these problems be tackled? (iii) What alternatives exist for solving a given problem? (iv) How should dependencies between problems be handled? (v) How to resolve each individual problem most effectively in the presence of its surrounding problems?

Considering that core ontologies provide a precise definition of structural knowledge in a specific field that spans across different application domains in this field [2], the knowledge they represent is to be reused in many domains. Thus, we argue that core ontologies are good candidates to be presented as ontology pattern languages. Therefore, aiming to favor reuse, we organized five core ontologies as OPLs [44]: Software Process OPL (SP-OPL), ISO-based Software Process OPL (ISP-OPL), Enterprise OPL (E-OPL), Measurement OPL (M-OPL), and Service OPL (S-OPL). Figure 7 shows a

fragment of SPO from which we identified patterns to create S-OPL. The description and models of the OPLs cited above can be found in [44].

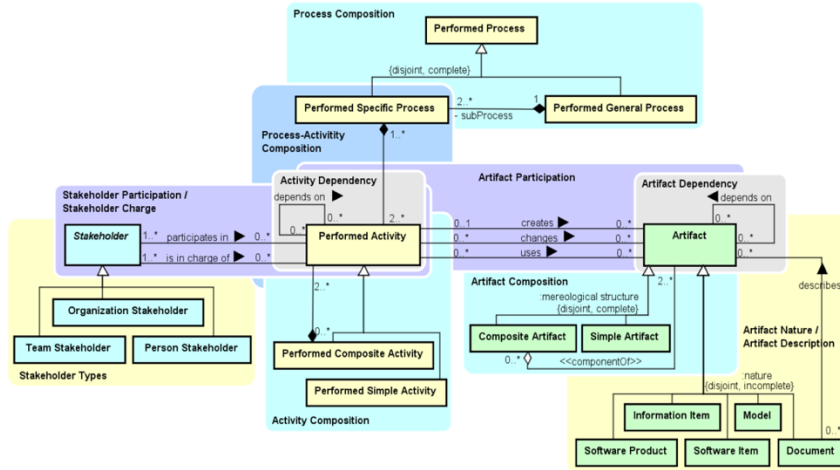


Fig. 7. Fragment of SPO with patterns identification [28].

When developing our OPLs, to create the OPLs diagrams that show how the patterns should be combined and the order in which they should be applied, we used an adaptation of the UML activity diagram. However, we experienced some limitations and there were inconsistencies in the notation we used in different OPLs. Thus, we noticed that it would be necessary to define a proper notation to represent OPLs. Hence, we created OPL-ML (*Ontology Pattern Language Modeling Language*) [45]. For developing OPL-ML, we relied on the results of a systematic mapping of the literature that investigated visual notations for Software Pattern Languages [46]. Moreover, OPL-ML was designed according to the principles of the Physics of Notation (PoN) [47], and following the design process defined by PoN-S (PoN Systematized), as we discuss in [48]. Figure 8 shows a fragment of the process (behavioral) model of S-OPL, which defines the flow to be followed to select the patterns to be applied according to the problem to be solved. Table 1 presents the notation used in the figure. The complete specification of S-OPL is available at <https://nemo.inf.ufes.br/projects/opl/>.

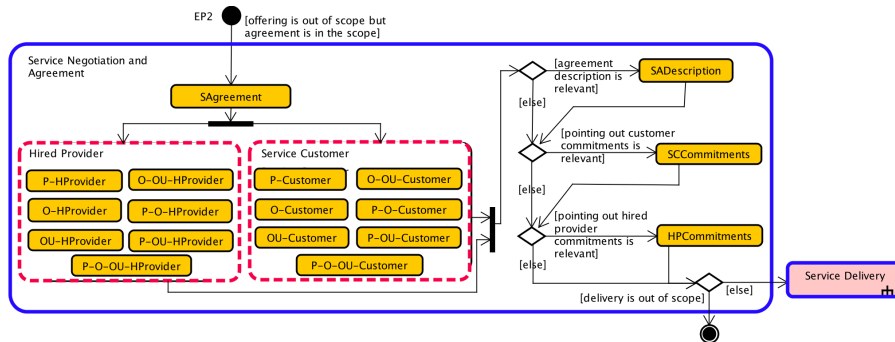
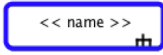
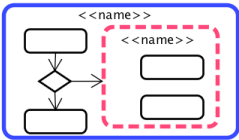




Fig. 8. Fragment of S-OPL – Behavioral Model [45].

**Table 1.** OPL-ML visual notation – Behavioral Model [45].

Element	Symbol
Pattern Application Action Group (black box format)	
Pattern Application Action Group (expanded format)	
Variant Pattern Application Action Group (black box format)	
Variant Application Action Group (expanded format)	

## 6 Personal Notes

This paper was written in honor of *Ricardo de Almeida Falbo*, on the occasion of his formal retirement. It has been more than ten years of partnership. In this paper, I talked about some works on which we worked together during this period. Falbo has contributed to the research topics addressed in this paper, and also to many others. More than that, he has contributed to people's life, in academical and personal sense, helping form better researchers, with better character. Many people (students, colleagues, partners) participated in the works cited in this paper, and I am sure Falbo made a difference in their lives. I can say that from my own experience.

In the path I have followed in the last years, Falbo has played an important role. I have experienced *Falbo's Approach* as his advisee, as partners advising students, as partners in research projects, and most of all, as a friend. We celebrated accepted papers, got frustrated because papers were rejected, discussed research issues, worked a lot, talked about life in general. It is so easy to work with him!

I was once told that we should be able to express our feelings for someone in just three words. In the case of Falbo, my words are *friendship*, *admiration* and *gratitude*.

In this paper, I reported works developed along the path followed in the last years. There is a road ahead. Currently, we are working on ontologies in the Human-Computer Interaction (HCI) domain. Some time ago, a student we advise told us that she was interested in investigating HCI aspects in her doctorate research project. HCI area was new for both of us. Even so, he accepted the challenge of getting into this area and invited me to this project. This has been our more recent adventure together, and we expect to produce interesting contributions to the HCI and Ontology communities. Let's do it, my dear friend!



## References

1. Guarino, N.: Formal Ontology and Information Systems. Formal Ontologies in Information Systems, IOS Press, 3 -15 (1998).
2. Scherp, A., Saathoff, C., Franz, T., Staab, S.: Designing core ontologies. *Applied Ontology*, 6(3), 177-221 (2011).
3. Guizzardi, G.: On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. In: Proceedings of the 2007 Conference on Databases and Information Systems IV: Seventh International Baltic Conference (DB&IS'2006), pp. 18-39 (2007).
4. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. Fundamental Research Series. Centre for Telematics and Information Technology. Netherlands (2005).
5. Guizzardi, G., Wagner, G., Falbo, R.A., Guizzardi, R.S.S, Almeida, J.P.A.: Towards Ontological Foundations for the Conceptual Modeling of Events. In: Proceedings of the 32th International Conference on Conceptual Modeling (ER 2013), pp. 327-341. Hong-Kong, China (2013).
6. Guizzardi, G., Falbo, R.A., Guizzardi, R.S.S.: 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 (2008).
7. Falbo, R.A., Barcellos, M.P., Nardi, J.C., Guizzardi, G.: Organizing Ontology Design Patterns as Ontology Pattern Languages. In: Cimiano P., Corcho O., Presutti V., Hol-link L., Rudolph S. (eds) *The Semantic Web: Semantics and Big Data*. 10th Extended Semantic Web Conference (ESWC 2013). Lecture Notes in Computer Science, vol 7882, pp. 61-75. Springer, Berlin, Heidelberg (2013).
8. Barcellos, M. P., Falbo, R. A., Rocha, A. R.: A Well-Founded Software Process Behavior Ontology to Support Business Goals Monitoring in High Maturity Software Organizations. In: Proceedings of the 14th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW). pp. 253-262. Vitória, Brazil (2010).
9. Barcellos, M. P., Falbo, R. A., Dalmoro, R.: A Well-Founded Software Measurement Ontology. In: 6th International Conference on Formal Ontology in Information Systems (FOIS 2010). Toronto - Canadá. *Frontiers in Artificial Intelligence and Applications*, 209, pp. 213-226. IOS Press, Amsterda (2010).
10. Barcellos, M. P., Falbo, R. A., Rocha, A. R.: Establishing a Well-Founded Conceptualization about Software Measurement in High Maturity Levels. In: Proceedings of the Seventh International Conference on the Quality of Information and Communications Technology (QUATIC). pp. 467-472. Porto, Portugal (2010).
11. Barcellos, M. P., Falbo, R. A., Rocha, A. R.: Using a Reference Domain Ontology for Developing a Software Measurement Strategy for High Maturity Organizations. In: Proceedings of the 16th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW). pp. 114-123. Beijing, China (2012).

12. Barcellos, M. P., Falbo, R. A., Rocha, A. R.: A strategy for preparing software organizations for statistical process control. *Journal of the Brazilian Computer Society*. v. 19, p. 1-31 (2013).
13. Barcellos, M. P., Falbo, R. A.: A software measurement task ontology. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC 2013)*. pp. 311-318. New York: ACM Press. Coimbra, Portugal (2013).
14. SEI, CMMI® for Development, Version 1.3, Pittsburg (2010).
15. Montoni M., Rocha A.R., Weber K.C.: MPS.BR: a successful program for software process improvement in Brazil. *Software Process Improve Practice*, 14, 289–300 (2009).
16. Calhau, R. F., Falbo, R.A.: An Ontology-Based Approach for Semantic Integration. In: *Proceedings of the 14th IEEE International Enterprise Distributed Object Computing Conference*. pp. 111–120. Vitória, Brazil (2010).
17. Izza, S.: Integration of industrial information systems: from syntactic to semantic integration approaches. *Enterprise Information System*, 3, 1–57 (2009).
18. Fonseca, V. S., Barcellos, M. P., Falbo, R.A.: An ontology-based approach for integrating tools supporting the software measurement process. *Science of Computer Programming*, 135, 20-44 (2016).
19. Renault, L. D. C., Barcellos, M. P., Falbo, R.A.: Using an Ontology-based Approach for Integrating Applications to support Software Processes. In: *Proceedings of the 17th Brazilian Symposium on Software Quality (SBQS)*. pp. 220-229. ACM International Proceedings Series. Curitiba, Brazil (2018).
20. Lethbridge, T. C., Singer, J., Forward, A.: How Software Engineers Use Documentation: The State of the Practice. *IEEE Software*, 20(6), 35 – 39 (2003).
21. Arantes, L., Falbo, R. A. An infrastructure for managing semantic documents. In: *Proceedings of the 14th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW)*. pp. 235-244. Vitória, Brazil (2010).
22. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American*. 284 (5), 34–43 (2001).
23. Sicilia, M.: Metadata, semantics and ontology: providing meaning to information resources. *International Journal of Metadata, Semantics and Ontologies*, 1(1), 83 – 86 (2006).
24. Eriksson, H.: The semantic-document approach to combining documents and ontologies. *International Journal of Human-Computer Studies*, 65(7), 624-639 (2007).
25. Bastos, E.C., Barcellos, M. P., Falbo, R.A.: Using Semantic Documentation to Support Software Project Management. *Journal on Data Semantics*. vol. 7(2), 107–132 (2018).
26. Talas, J., Gregar, T., Pitner, T.: Semantic wiki in environmental project management. In: *IFIP Advances in Information and Communication Technology*. pp. 437-444. Brno, Czech Republic (2011).
27. Ruy, F. B., Falbo, R.A., Barcellos, M.P., Guizzardi, G.: An Ontological Analysis of the ISO/IEC 24744 Metamodel, In: *8th International Conference on Formal Ontology in Information Systems (FOIS 2014)*. Rio de Janeiro, Brazil. *Frontiers in Artificial Intelligence and Applications*, 267, pp. 330-343. IOS Press, Amsterda (2014).
28. Ruy, F. B.: Software Engineering Standards Harmonization: An Ontology-Based Approach. Doctoral Thesis. Postgraduate Program in Computer Science. Federal University of Espírito Santo, Brazil (2017).

29. Ruy, F. B., Falbo, R.A., Barcellos, M.P., Costa, S.D., Guizzardi, G.: SEON: A Software Engineering Ontology Network. In: Blomqvist E., Ciancarini P., Poggi F., Vitali F. (eds) Knowledge Engineering and Knowledge Management. EKAW 2016. Lecture Notes in Computer Science, vol 10024, pp. 527-542. Springer, Cham (2016).
30. Ruy, F. B., Souza, E., Falbo, R.A., Barcellos, M.P.: Software Testing Processes in ISO Standards: How to Harmonize Them? In: Proceedings of the 16th Brazilian Symposium on Software Quality (SBQS). pp. 296-310. Rio de Janeiro, Brazil (2017).
31. Briguento, A. C. O., Falbo, R. A., Guizzardi, G.: Using a Foundational Ontology for Reengineering a Software Process Ontology. In: Proceedings of the 16h Brazilian Symposium on Data Base. São Paulo, Brazil (2012).
32. Peçanha, C.C., Duarte, B.B., Souza, V.E.S.: RASO: an Ontology on Requirements for the Development of Adaptive Systems. In: Proceedings of the 21st Workshop on Requirements Engineering (WER 2018). pp. 1–14. Rio de Janeiro, Brazil (2018).
33. Arantes, L.O., Falbo, R.A., Guizzardi, G.: Evolving a Software Configuration Ontology. In Proceedings of the Second Brazilian Workshop on Ontologies and Metamodels for Software and Data Engineering (WOMSDE07). João Pessoa, Brazil (2007).
34. Souza, E.F.D., Falbo, R.A., Vijaykumar, N.L.: ROoST: Reference Ontology on Software Testing. *Applied Ontology*,12, 59–90 (2017).
35. Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A.: *Ontology Engineering in a Networked World*. Springer Science & Business Media (2012).
36. d'Aquin, M., Gangemi, A.: Is there beauty in ontologies? *Applied Ontology*, 6 (3), 165–175 (2011).
37. Falbo, R.A., Ruy, F. B., Guizzardi, G., Barcellos, M.P., Almeida, J.P.A.: Towards an Enterprise Ontology Pattern Language. In: Proceedings of the 29th ACM Symposium on Applied Computing (ACM SAC). pp. 323-330. Gyeongju, Republic of Korea (2014).
38. Barcellos, M.P., Falbo, R.A., Frauches, V.G.V.: Towards a Measurement Ontology Pattern Language. In: Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering. CEUR Workshop Proceedings. Vol. 1301 (2014).
39. Falbo, R.A.: SABiO: Systematic Approach for Building Ontologies. In: Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering. CEUR Workshop Proceedings. Vol. 1301 (2014).
40. Villalon, M.P., Suárez-Figueroa, M. C., Gómez-Pérez, A.: Reusing ontology design patterns in a context ontology network. In: 2nd International Workshop on Ontology Patterns (WOP 2010). CEUR Workshop Proceedings, Vol. 671, pp. 35-49 (2010).
41. Presutti, V., Daga, E., Gangemi, A., Blomqvist, E.: eXtreme Design with Content Ontology Design Patterns. In: Workshop on Ontology Patterns (WOP). CEUR Workshop Proceedings, Vol. 516, pp. 83-97 (2009).
42. Deutsch, P.: *Models and Patterns*. In: *Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools*, Greenfield, J., Short, K., Cook, S., Kent, S. (eds.), John Wiley & Sons (2004).
43. Buschmann, F., Henney, K., Schmidt, D.C.: *Pattern-Oriented Software Architecture: On Patterns and Pattern Languages*, John Wiley & Sons Ltd (2007).

44. Falbo, R.A., Barcellos, M.P., Ruy, F.B., Guizzardi, G., Guizzardi, R.S.S.: *Ontology Pattern Languages*. In: Gangemi, A., Hitzler, P., Janowicz, K., Krisnadhi, A., and Pre-sutti, V., editors, *Ontology Engineering with Ontology Design Patterns: Foundations and Applications*. IOS Press (2016).
45. Quirino, G., Barcellos, M.P., Falbo, R.A.: *OPL-ML: A Modeling Language for Representing Ontology Pattern Languages*. In: de Cesare S., Frank U. (eds) *Advances in Conceptual Modeling. ER 2017. Lecture Notes in Computer Science*, vol 10651, pp. 187-201. Springer, Cham (2017).
46. Quirino, G., Barcellos, M.P., Falbo, R.A.: *Visual Notations for Software Pattern Languages: a Mapping Study*. In: *32nd Brazilian Symposium on Software Engineering (SBES)*. pp. 72-81. ACM International Proceedings Series. São Carlos, Brazil (2018).
47. Moody, D.L.: *The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering*. *IEEE Transactions on Software Engineering*. 35(6), 1–22 (2009).
48. Teixeira, M. G. S., Quirino, G., Gailly, F., Falbo, R. A., Guizzardi, G., Barcellos, M. P.: *PoN-S: A Systematic Approach for Applying the Physics of Notation (PoN)*. In: *Proceedings of the 21st International Conference in Exploring Modelling Methods for Systems Analysis and Design (EMMSAD 2016)*. pp. 432–447. Ljubljana, Slovenia (2016).