Towards an ontology modeling tool. A validation in software engineering scenarios

Francisco José García-Peñalvo, Universidad de Salamanca, SPAIN

Ricardo Colomo-Palacios, Universidad Carlos III de Madrid, SPAIN

Juan García, Universidad de Salamanca, SPAIN

Roberto Therón, Universidad de Salamanca, SPAIN

Abstract

Ontology creation and management related processes are very important to define and develop semantic services. Ontology Engineering is the research field that provides the mechanisms to manage the life cycle of the ontologies. However, the process of building ontologies can be tedious and sometimes exhaustive. OWL-VisMod is a tool designed for developing ontological engineering based on visual analytics conceptual modeling for OWL ontologies life cycle management, supporting both creation and understanding tasks. This paper is devoted to evaluate OWL-VisMod through a set of defined tasks. The same tasks also will be done with the most known tool in Ontology Engineering, Protégé, in order to compare the obtained results and be able to know how is OWL-VisMod perceived for the expert users. The comparison shows that both tools have similar acceptation scores, but OWL-VisMod presents better feelings regarding user's perception tasks due to the visual analytics influence.

Keywords

OWL, Ontologies, Knowledge Engineering, Visual Analytics, Software Engineering

Introduction

Semantic Technologies are one of the fastest developing fields within the Information and Communication Technology sector and, as such, under constant examination by scientists and IT professionals (Janev & Vranes, 2011). Semantic, from the Greek "sēmantikos", involves giving significance or meaning to words or symbols, enabling distinctions between the meanings of different words or symbols. Semantic technologies are based on ontologies (Fensel, 2002). Ontology formalizes knowledge meaning and facilitates the search for contents and information (Jiang & Tan, 2009). The main objective of ontologies is to establish ontological agreements, which serve as the basis for communication between either human or software agents, hence, reducing language ambiguity and knowledge differences between agents, which may lead to errors, misunderstandings and inefficiencies (Blanco et al., 2011).

Now, semantic technology research relies on a number of key methodologies such as knowledge representation languages or reasoning algorithms (Hitzler & Janowicz, 2011). The application of ontologies for expressing semantics of data does not restrict any longer exclusively on semantic web or semantic web services (Vrba et al., 2011).

According to Breslin et al. (2010), industry has begun to watch developments with interest and a number of large companies have started to experiment with Semantic technologies to ascertain if these new technologies can be leveraged to add more value for their customers or internally within the company, while there are already several offers of vendors of Semantic solutions on the market. Due to this expansion several fields has been affected by semantics and many solutions and initiatives have been developed. Software Engineering is one of them. As a result of this there are many initiatives reported in the literature that employ semantic technologies in aspects like requirements (Chicaiza et al., 2010), analysis (Tappolet, Kiefer & Bernstein (2010), modeling (Gallardo et al., 2011; Martinho, Varajao & Domingos (2010); Sicilia et al., 2009), teaming (Soto-Acosta et al., 2010; Valencia-García et al., 2010), cooperative building (Tacla et al., 2011), software metrics (García-Crespo et al., 2009), reuse (Shiva & Shala, 2008) or quality management (García et al., 2010) to cite some of the most relevant and recent cases.

Ontologies represent one of the most common representations of the semantic technologies (García-Peñalvo, García, & Therón, 2011). There is a research field called Ontology Engineering, which provides the mechanisms to manage the life cycle of them. The Ontology Engineering has been described as an investigation methodology that provides the rational design of a knowledge base (Mizoguchi, 2004). It also provides the principles for the set of activities and processes that cover the life cycle of ontologies. The main of these processes are the creation, management, analysis and reuse of ontologies.

As well as the processes, the Ontology Engineering also covers other aspects such as metrics, methodologies and the diverse tools for creating, editing and visualizing ontologies. Most of these ontology editors and tools are based on the use of simple visualizations, having diverse problems, as has been widely documented (e.g. García, Therón & García-Peñalvo, 2011; García, García-Peñalvo & Therón, 2011). These problems are mainly the occlusion of visual elements, the overcrowded visualizations, a lack of robust interaction techniques and a poor implementation of the visual expressivity, a concept defined as the number of visual variables used for enriching visualizations (Ware, 2004).

A solution to these visualization problems is the use of Visual Analytics techniques. Visual Analytics is a multidisciplinary research field focused on the development of diverse analytical reasoning techniques, visual representations and interaction techniques, combined with a set of data representations and transformations. It has been more formally defined as: Visual analytics is the science of analytical reasoning facilitated by interactive visual interfaces (Thomas & Cook, 2005).

In the Visual Analytics field, the user represents the main aspect in the process of analysis. He develops the analysis and the tools support this process. It is crucial the development of robust tools and visual and interactive techniques that support this analysis. This field is based on the use of the human cognitive capacities enriched with the currently computer capabilities. The result is a set of robust tools that the user can use to analyze information, and based on this analysis, first, to get knowledge from the data model and second, to take decisions or to execute diverse actions.

Visual Analytics has been used in diverse research domains, such as Bioinformatics (Baehrecke et al., 2004), Geography (Andrienko et al., 2007) or Medicine (Tominski et al., 2008). Moreover, the industry it is also taking advantage in diverse fields such as databases (Shneiderman, 2008), Software Engineering (Isenberg and Fisher, 2009; Telea & Voinea, 2009) or the pharmacy (Saffer et al., 2004). Nevertheless, there is no any antecedent of the use of Visual Analytics in the field of Ontological Engineering (e.g. Gómez-Pérez et al., 2003).

The advantages of using a Visual Analytics approach to develop the Ontological Engineering are diverse. The first advantage is that the use of robust visualization techniques, let to discover new knowledge of the ontologies, specially, during an analysis phase for reusing.

A second advantage is that the visual modeling process of creating ontologies becomes easier than the use of traditional ontologies editors based on widgets such as comboboxes, textfields, etc. Without any doubt, the use of visualizations improves the cognitive process to analyze an ontological model.

This paper is focused on providing a validation of the OWL-VisMod tool, which aims to contribute to the development of Ontological Engineering, the branch of Knowledge Engineering that exploits the formal principles to build ontologies. The main purpose behind OWL-VisMod is to provide users with a tool to support the development, creation, management, maintenance and reusability of OWL ontologies for Knowledge-based systems (García, García-Peñalvo, & Therón, 2010a; García, García-Peñalvo, & Therón, 2010b). The usability of OWL-VisMod has been evaluated by means of an empirical study, with good results (García, García-Peñalvo, Therón & de Pablos, 2011).

The paper consists of four sections and is structured as follows. Section 2 reviews the relevant literature about the field of study of OWL-VisMod. Section 3 describes the tool paying attention to its architecture and main features. Section 4 describes the evaluation process carried out. Finally, the paper ends with a discussion of research findings, limitations and concluding remarks.

Literature Review

The main processes involved in the life cycle of ontologies are the creation, maintenance, analysis and reuse. The creation process consists of activities and workflows that have been defined in diverse methodologies. Uschold and King (1995) proposed one of the first methodologies specially focused on the creation process, called Knowledge Engineering Methodology (KEM). This proposal describes some of the most important tasks, involved in the process of the creation of ontologies. Figure 1 illustrates the most important activities defined in the KEM Methodology. It starts with the definition and conceptualization of the domain, followed by an analysis phase in order to reuse existing ontologies in the model that is being built. Then, the formal specification of the ontology includes the definition of the taxonomy of concepts, the attributes and relations. Once the ontology has been built, the next phase involves the creation of the individuals or instances that populate the ontology, to finally conclude with the evaluation and documentation processes.

Another relevant methodology that has been taken as base for future proposals is Methontology (e.g. Fernández-López, Gómez-Pérez & Juristo 1997). Methontology covers the whole life cycle of ontologies, and includes a tool called WebODE that supports all the activities defined on it.

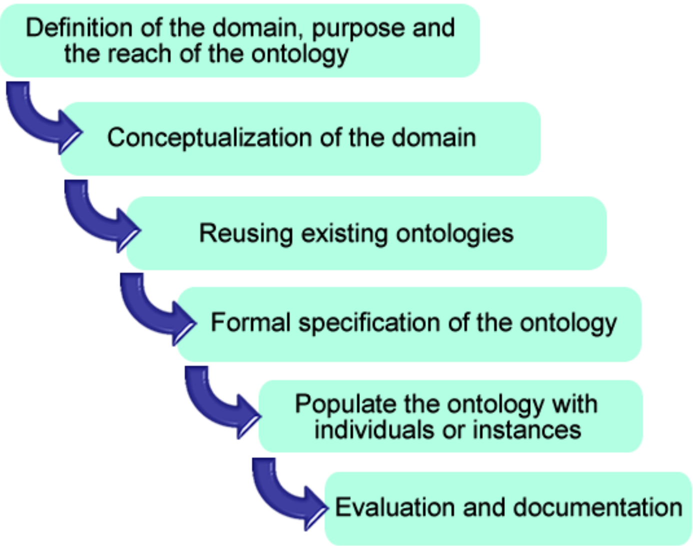


Figure 1 Six defined phases in the methodology *Knowledge Engineering Methodology (KEM).*

Methontology is focused on the development of ontologies from the level of knowledge, through an approach close to the traditional cascade process defined in the Software Engineering field. This proposal defines four phases to build an ontology: the first phase is the definition of the reach and the granularity, the second phase is the conceptualization of the domain, the third phase is the implementation of the ontology in a language such as RDF or OWL. Finally, the fourth phase is the evaluation of the ontology.

DOGMA (Development of Ontology Guided Methodology Approach) is a framework for developing the Ontology Engineering in a very formal manner (Jarrar & Meersman, 2002). The philosophy behind DOGMA is the reuse of ontologies, due to they are considered as scalable and shared resources that let to reuse the knowledge (Jarrar & Meersman, 2008). The reuse of the ontologies is due to the methodology proposes the definition of diverse levels of abstraction, starting from an upper level with very general concepts, that can let these models to be reused in diverse domains.

Apart from the methodologies, the Ontology Engineering also requires tools that support all the activities defined in the processes. Diverse tools have been designed (Suresh, Kumar, Prakash & Rizvi, 2008), nevertheless, all these proposals do not support methodologies. In contrast, they are independent proposals, except for Methontology and DOGMA that have implemented specific tools that support the activities defined.

Diverse commercial tools have been proposed for modeling ontologies. The most important currently are: SemanticWorks, TopBraidComposer and OntoStudio. Some of these tools offer a free version with reduced funcionality. There are other free to use tools to model an edit ontologies such as NeOn Toolkit (Haase et al., 2008), OntoEdit (Sure, Angele & Staab, 2002), HOZO (Kozaki et al., 2005; Sunagawa et al., 2005), but Protégé (Gennari et al., 2003) is the most widely used tool for editing ontologies.

SemanticWorks1 is a commercial tool designed to edit RDF documents in a GUI and check its sintaxis, as well as design RDF schema and OWL ontologies using a graphical design View, based on a conceptual map approach. It checks the syntaxes and semantics of ontologies, using a graph modeler based on the use of conceptual maps.

The European Union has visualized the potential of the development of the diverse semantic technologies, and has supported research in this direction. The project NeOn[2] can be the most important proposal in this area. Its main goal is to manage multiple ontologies in a specific context, that are created as result of a collaboration of diverse entities, and can be dynamic and under evolution.

As a result of this research project, diverse tools and applications ontology-based have been released (e.g. Suárez-Figueroa et al., 2007; Villalón-Terrazas et al., 2011). One of the most important tools is an environment for developing the Ontology Engineering called NeOn Toolkit (Haase et al., 2008). This robust platform is open to new developments of Eclipse plugins that can be added to the toolkit.

TopBraid Composer[3] illustrated on figure 2 is an enterprise class modeling environment for developing Semantic Web ontologies and building semantic applications. There are three available versions: a Free Edition, Standard Edition and Maestro Edition. TopBraid Composer is a UML-based modeling plug-in eclipse, part of the TopBraid Suite. We tested using TopBraid Composer Free Edition version 3.3.0 which does not support the UML representation that is provided only with paid versions. TopBraid Composer is a fully Protege-based tool that performs the most common operations over ontologies, such as: inference, consistency checking, and the inclusion of SPARQL query engine

[1] http://www.altova.com/semanticworks/owl-editor.html
[2] http://www.neon-project.org/
[3] http://www.topquadrant.com/

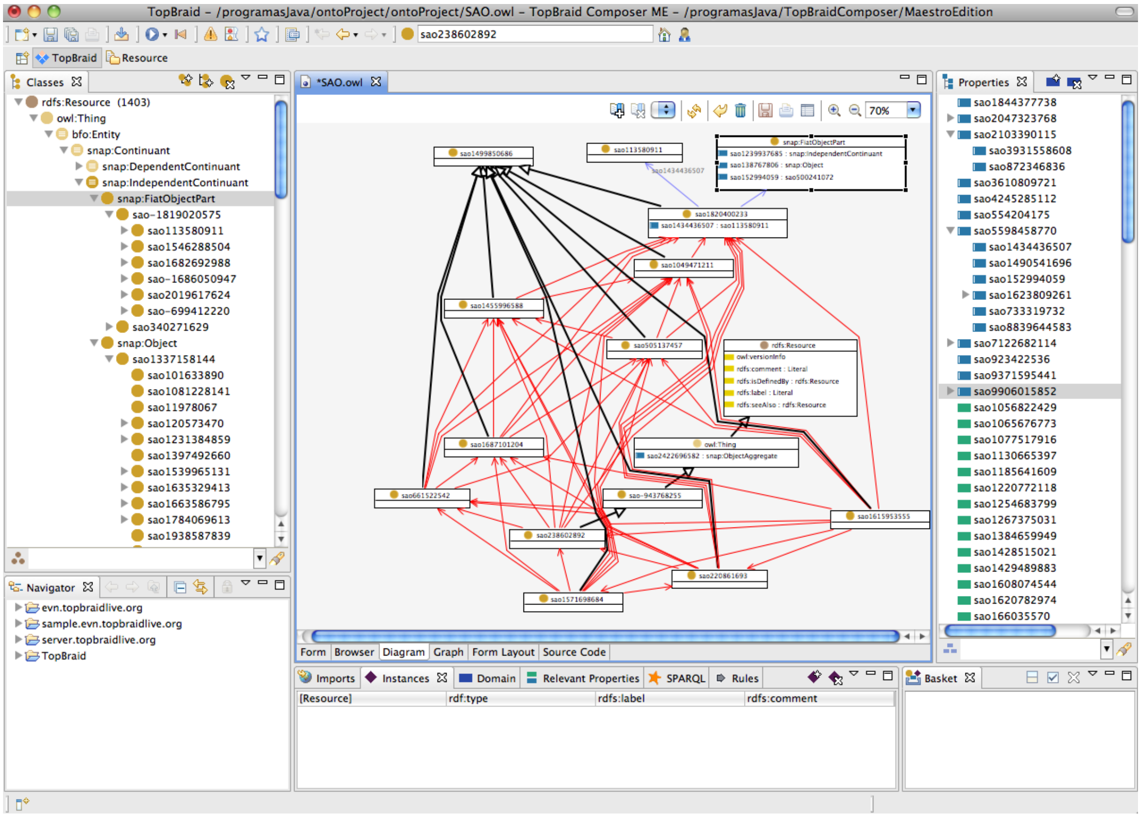


Figure 2 TopBraid Composer uses a UML-based visualization to represent both the hierarchy and the relations among classes.

Protégé (Gennari et al., 2003) is a free, open source ontology editor and knowledge-based framework. Protégé includes diverse plugins developed and maintained by the community. One of these plug-ins is OWLViz, a graph-based visualization that represents classes, properties, hierarchy, and the classical tree of hierarchies view. Classes are represented as nodes in the graph, while properties are represented as edges connecting nodes, where the edges represent "is-a" relationships (hierarchy).

Jambalaya (Storey et al., 2001) is another plug-in intended to visualize OWL ontologies with Protégé. It is a visualization tool not provided with modeling capabilities. Jambalaya is a complete plug-in that visually represents the components of the ontology and its relationships divided into two views. Each view can be displayed using one of six different layouts: grid, radial, spring, sugiyama, tree and treemap. This tool offers a great variety of configuration options - hiding components, changing colors and shapes and filtering data. Although Jambalaya represents a very good tool to visualize an ontology, the scalability is the main disadvantage due to the fact that large graph visualizations are well known to become cluttered.

Katifori et al. (2007) provided with a classification of the diverse tools for visualizing ontologies. They defined six categories according to the different characteristics of the presentation, interaction, technique, functionality supported or visualization dimensions. Nevertheless most of the tools fall in more than one category. There are two groups not included in this category; the first group is based on those tools that use UML notation to model ontologies, and the second group is formed by those tools based on the use of conceptual maps.

These eight groups are: indented lists, node-link (graphs) and trees, zoomables, space-filling, focus+context or distortion, 3D information landscapes, the UML-based and the conceptual maps-based. Basically, it can be distinguished two main modeling approaches for representing the relationships among classes: the first one using the well-known graph theory (like Protégé and SemanticWorks) and the second approach using UML diagrams, such as in the Object Oriented approach (TopBraid Composer).

All the tools based on the use of graphs (node-link) share the same problems. The first is the lack of a layout, and the majority of the time the user has to manually move the visual elements to organize them. The second problem is the scalability. It is well known that graphs are not good to represent a large amount of elements; these problems are illustrated on Figures 3 and 4. The tools based on UML practically have the same problems which are illustrated on Figure 2, even when this figure solely visualizes less than twenty classes.

As a result of an analysis of the current diverse tools, we have identified the main problems, including: the symbol redundancy (SemanticWorks), overcrowding of visual elements that difficult the understanding of visualizations, such as those based on directed graphs or UML. This problem is caused due to the majority of the tools saturating the visualizations and putting together the taxonomy with relationships. Therein the user gets easily confused and lost navigating the visualization. Another detected problem is the lack of layout, in the case of graphs and UML diagrams. A lack of a layout makes it difficult to find elements, and create a conceptual map of the knowledge base that is represented.

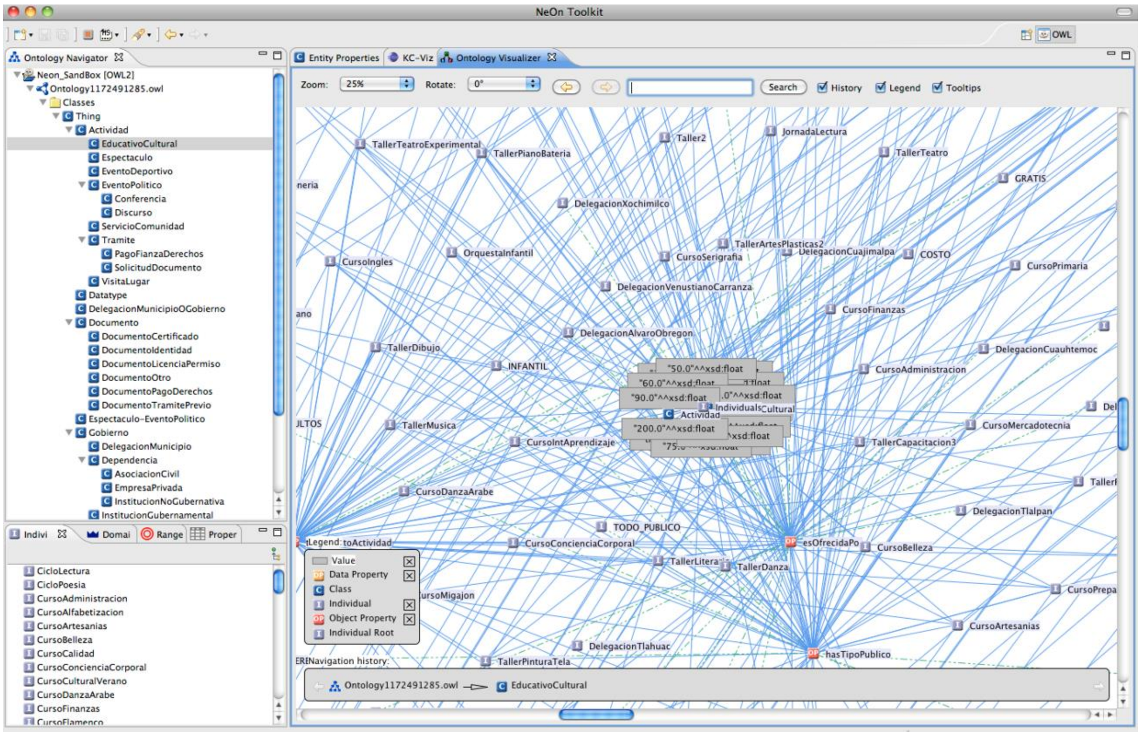


Figure 3 The NeOn Toolkit illustrates the problems of oclussion and overcrowding of visual elements.

Figure 3 illustrates the main problems that share the most of the visualizations tool. First, due to most of them use basic visualization techniques such as directed graphs, as shown on Figure

3. One of the problems is that visualizations display all the information in the same views, such as properties, classes or individuals. This strategy causes that as can be seen on Figure 3, the visual elements overlap others, the edges cannot be followed, and the visualization becomes completely overcrowded.

Most of the tools share the same problems, due to as can be seen on Figure 2 with TopBraidComposer and Figure 3 with NeOn Toolkit, most of them have decided to follow the Protégé's approach, sharing its same problems. Figure 4 shows the Protégé TGVizTab view, which is based on the use of directed graphs. It can be seen that the three tools described, share exactly the same problematic. Moreover, other tools described in (García, García-Peñalvo & Therón, 2011) also share the same problematic.

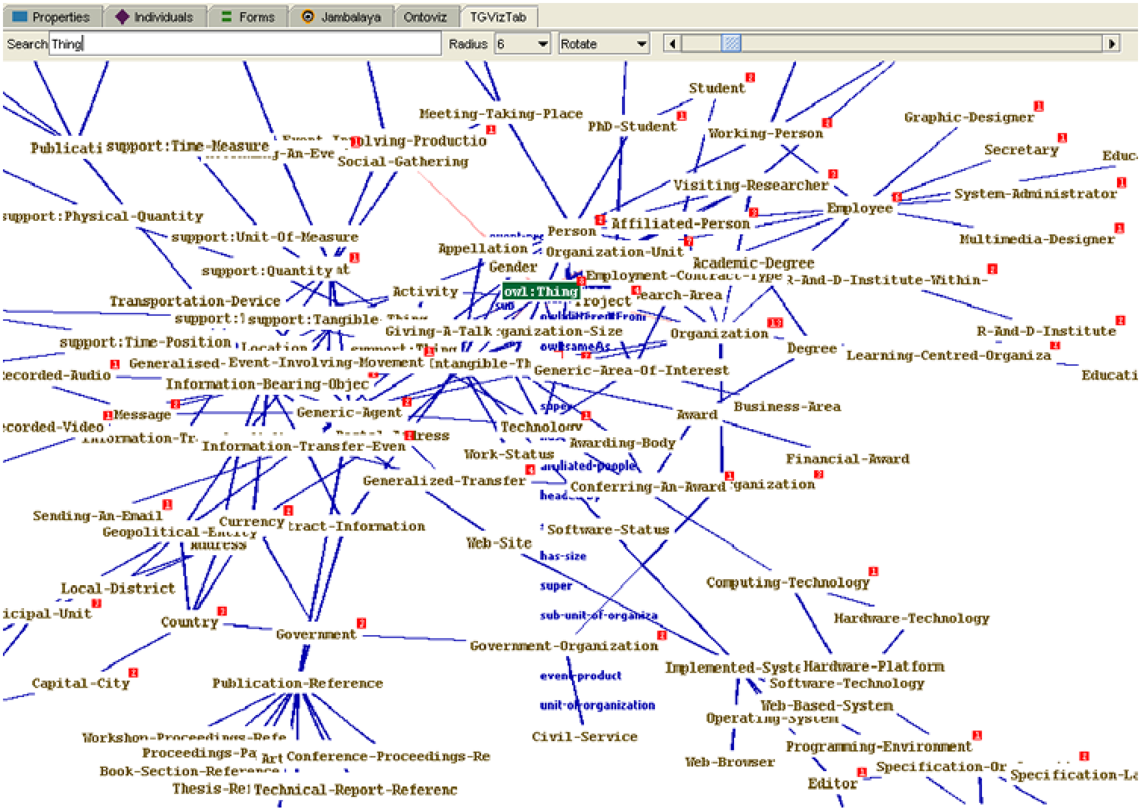


Figure 4 shows the Protégé TGViztab visualization based on the use of directed graphs. The main problems with this visualization are the occlusion and the overcrowding of visual elements.

The Tool

The analysis started with the loading of the ontology[4] in OWL-VisMod. Then, the first activity consisted on the navigation of the taxonomy of concepts, to detect those related to software safety. The taxonomy of concepts is defined in OWL-VisMod using two visualization techniques: the treemap and the hierarchical tree.

[4] http://www.seontology.org/permit_on/src/genericOnto.owl

The first technique is the treemap (Johnson & Shneiderman, 1991), a widely used technique for representing especially, large hierarchies (Bederson et al., 2002). This visualization technique is based on the efficient use of the whole available visual space in the dimensional plane. It is based on the use of two-dimensionally squared maps, where the lower levels are represented as internal squares located inside the higher level maps.

Figure 5 shows a treemap view representing the taxonomy of the SEOntology. This analysis is focused on those classes related to software security, so the class called “Software_Design” has been highlighted in order to analyze its subclasses. The total of classes in the ontology is 365, with 180 datatype properties, and 129 object properties.

The main classes are subclasses of the class “Software_Engineering_Domain”, and these classes are: Software_Design (highlighted in Figure 5), Software-Testing, Software_Construction, Software_Tools and Software_Requirements. All these classes related to the different processes involved in a software developing process are shown in the Figure 5. These classes represent the main aspects involved in the life cycle of a software project.

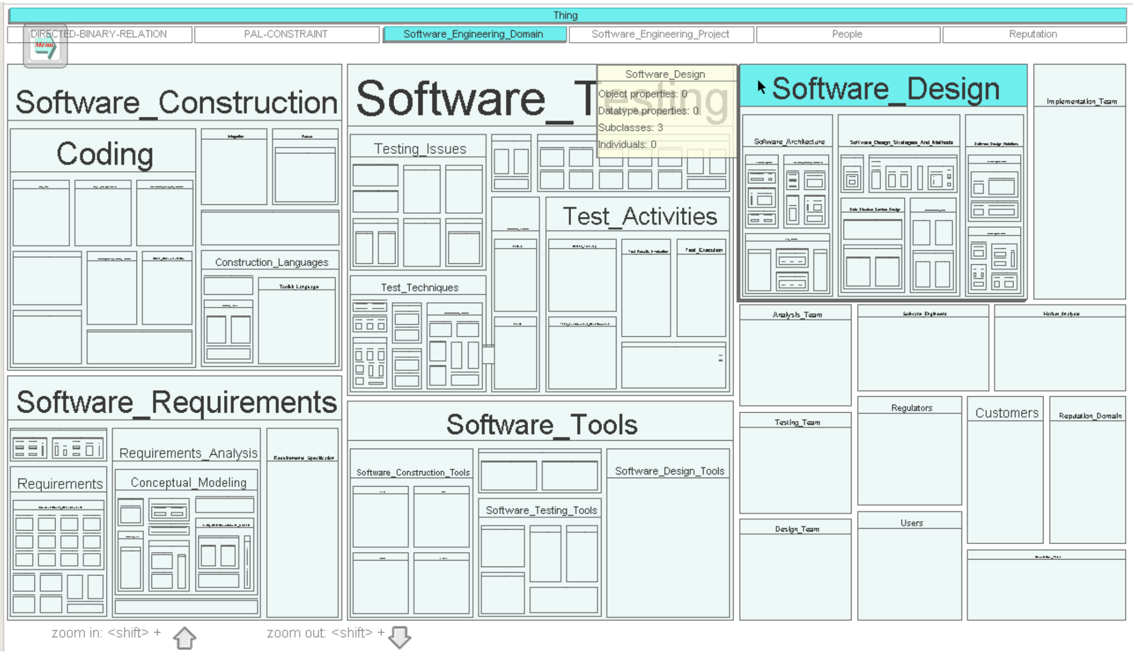


Figure 5 The treemap represents the hierarchy of the ontology SEOntology. The class "Software_Design" has been highlighted, and its superclass “Software_Engineering_domain” is also highlighted.

The second visualization technique that is used to analyze the taxonomy of concepts is the hierarchical tree. This visualization is a complementary view of the treemap, it uses the representation model of edges that connect the nodes, representing the elements (Tominski et al., 2006).

The analysis to the taxonomy of concepts, let to identify that there are no specific security-related concepts in this ontology. Diverse aspects related to the software engineering project have been defined, nevertheless, the problem of security and the safety software has not been considered in this ontological model. There are just two general concepts involved with security: the class “Safety_Requirements” and the class “Safety-critical_Systems_Testing”. Nevertheless, neither of them has defined object properties nor subclasses. Figure 6 (a) shows the datatype properties of the class *Safety_Requirements*, in this case five properties. On the other hand the class *Safety-critical_Systems_Testing* has only one datatype property defined.

This semantic zoom representation is based on a UML-like view, having an internal graph with a radial layout (García, Therón & García-Peñalvo 2011). Colors are used to indicate the type of the property or individuals. More specifically, the red is used to represent object properties, the green is used to represent the datatype properties, while the purple is used to represent the individuals of a class.

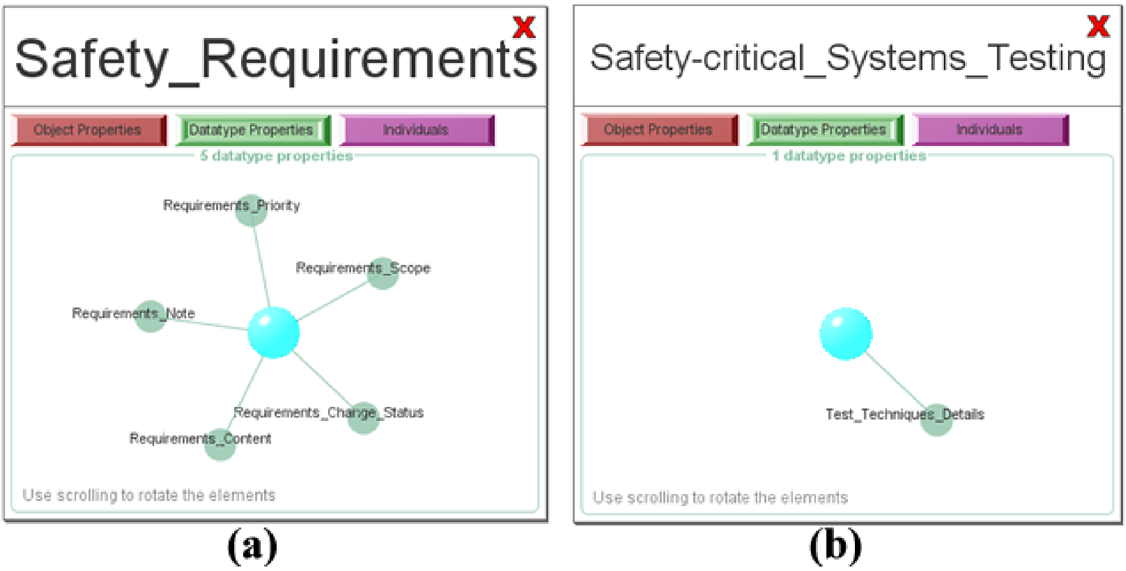


Figure 6 (a) shows the datatype properties of the class *Safety_Requirements*, while the Figure (b) shows the datatype properties of the class *Safety-critical_Systems_Testing*.

Figures 6 (a) and 6 (b) use the semantic zoom visualization technique to display the details of a specific class in the ontology. This technique is based on showing details according to the user's needs, changing the type and the meaning of the displayed information. Its main advantage is that the global context can be remained, while a detailed view of a certain element is shown (Herman et al., 2000).

OWL-VisMod has implemented a visualization technique to represent the global coupling of an ontology (García et al., 2011). This visualization shown on Figure 8 is based on a radial layout of the coupled classes in the ontology. Relations among classes also called coupling relationships are defined using Bezier curves in the same way, called Hierarchical Edge Bundles (Holten, 2006). This type of edges can be "tensed" to get more clear visualizations, avoiding the occlusion of edges. The use of a color varying from a tone to other, indicates the direction of the property, avoiding the use of arrowheads that overcrowd the visualization with elements. For the specific case of OWL-VisMod, the object properties are represented with a curve varying from red to yellow, while the datatype properties are represented with a curve varying from green to yellow.

Another visualization technique that has been implemented is the coupling of a specific class (García et al., 2010b), according to the coupling metrics defined in (García et al., 2010a). This visualization has been implemented in order to represent the semantic meaning of the coupling among classes. This coupling is interpreted as a relationship between two classes, where this relationship is defined as a mathematical function, having a class in its domain, and a class in its range. This interpretation of the semantic will be explained in detail with an example, in the next section.

According to the result of this analysis, the first action that should be taken, is the creation of diverse concepts related to safety mechanisms, in order to enrich this ontological model. Then, for each new class created, diverse properties should be defined, to finally populate the ontological model with instances of the class. These activities are described in detail in the following section.

Tool Evaluation

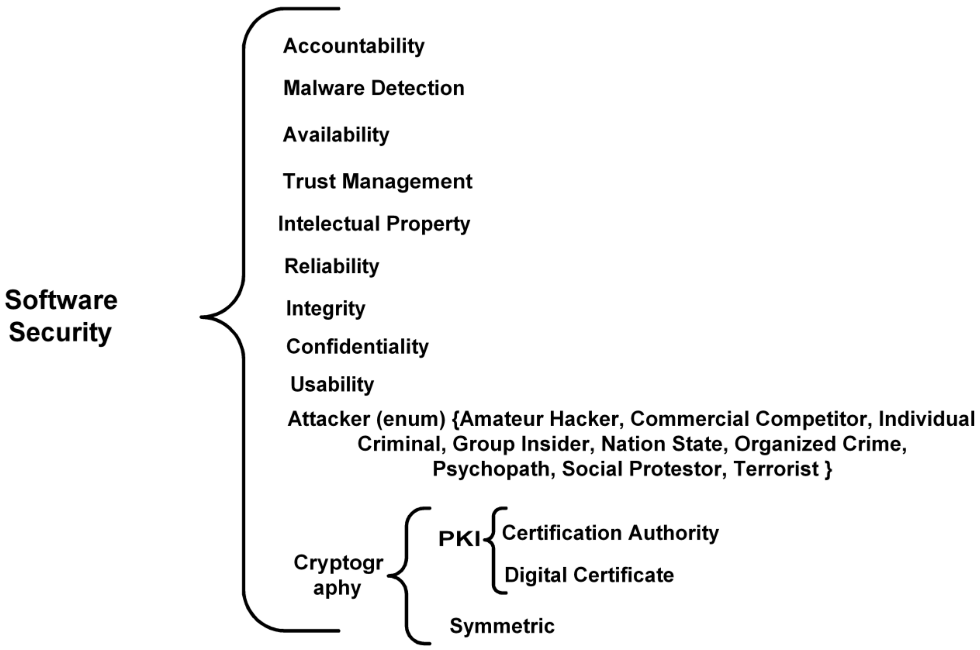
With the aim of getting feedback concerning the tool compared with well-known solutions, an evaluation was carried out as described in the subsequent paragraphs.

Experimental Design

The evaluation of the system consists in the performance of a set of tasks by a set of subjects. This set of tasks will be performed using a Domain Ontology devoted to Software Engineering, in this case Software engineering ontology (SEOntology), and performed using two different tools. The first set of tasks will be performed using OWL-VisMod and the second one using Protégé. After these tasks, all users are asked to answer a questionnaire. The final aim is to compare the results of both questionnaires in order to set the validity of OWL-VisMod compared with a recognized tool like Protégé.

The task is divided in several steps described as follows:

1. Load SEOntology
2. Navigate the taxonomy looking for specific concepts. In this case, related to Software Safety.
3. Analysis of Safety_Requirements and Safety-critical_Systems_Testing classes (both related to Software Safety), checking that both are defined as isolated classes.
4. Define new classes as follows:



5. Define properties as follows:

Class	Type	Name
Malware_Detection	datatype (String)	detection_mechanism
Malware_Detection	datatype (String)	malware_name
Intelectual_Property	datatype (boolean, functional)	Registered
Trust_Management	Object (Certification Authority)	certification_authority
Trust_Management	datatype (boolean, functional)	uses_certificate
Trust Management	Object(Digital Certificate)	digital_certificate
Reliability	datatype (String)	identity_management
Usability	datatype (String)	Log
Confidenciality	datatype (String)	access_control
Confidenciality	object (Symmetric)	cipher_algorithm
Integrity	datatype (String)	hash_algorithm

6. Define instances as follows:

Classes	Instances
Certification_Authority	VerySign, CERES, GVA
Digital_Certificate	VerySignCertificate, CERES_Certificate, GVA_Certificate
Malware_detection	detection_mechanism= static analysis based
	detection_mechanism= code graphs based
Symmetric	DES, 3DES, AES, Blowfish

7. Save the ontology

Once subjects performed the tasks they were asked to answer a questionnaire. This questionnaire consisted on a set of questions that evaluate the performance of OWL-VisMod and Protégé using the same set of questions by means of a Likert scale (1 to 10 points with anchors from very poor to very satisfactory):

- Rate taxonomic visualization of the classes that this tool provides.
- Rate the visual analytics process of the tool.
- Rate the ontology modeling workflow.
- Rate the user experience of the tool.
- Rate the global performance of the tool.

A pilot application was made prior to the final implementation of the questionnaire. The sample for this pilot implementation was composed by three semantic technologies experts. The objective of this pilot study was the improvement and assurance of the associated documentation. This resulted in several changes in formats and tables in the wording of some texts.

Data collection was conducted through a questionnaire that obtained information from the sample. All questionnaires were filled out by subjects with the assistance of at least one researcher. Questionnaires were answered on printed copies and subsequently coded in the statistical analysis tool GNU R.

Sample

The study was carried out over a period of two weeks. Participants were obtained from those who responded positively to a personal invitation, the sample consisted of 21 subjects, 4 women and 17 men. The average age was 27.9. In average, subjects have 2.3 years of experience in the field of semantic technologies.

Threats to Validity

In this study internal or external validity threats are present. With respect to the first, the respondents may not have a comparable level of knowledge or expertise. However, sample was chosen because of their expertise and experience, authors made sure that experts possessed a comparable level of knowledge and expertise.

The Figure 7 depicts the tree visualization after the creation of the concepts related to security. The class *Software_Security* has been highlighted in a red colored rectangle, as well as its superclass *Software_Engineering_Domain*. This class is defined as an upper class for diverse concepts related to the security of the information, such as *Confidentiality*, *Usability*, *Availability*, *Reliability* or *Trust Management* among others, shown in the Figure 7. In total, fourteen new concepts have been created to the ontological model, as previously has been described.

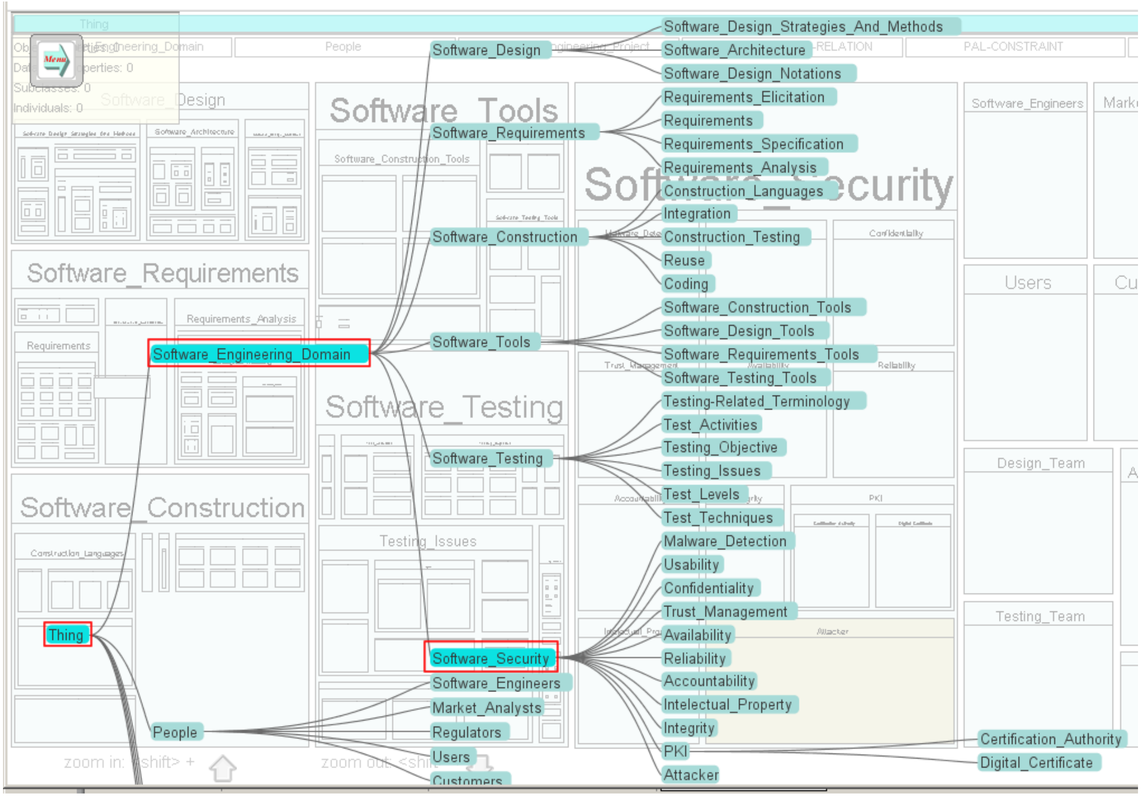


Figure 7 The tree of hirarchies visualization shows the taxonomy of concepts of SEOntology. The class "Software_Security" has been highlighted, as well as its superclasses.

Once the new concepts have been created in the ontology, the next phase is the definition of the relations among classes. Figure 8 shows the relations of the class *Trust_Management* with the classes *Digital_Certificate* and *Certification_Authority*. These relationships indicate that in order to have a trusty relation between two parts, there is a need of having a digital certificate validated by a certification authority.

The visualization technique shown in Figure 8, highlights the focused class (*Trust_Management*) as well as the coupled classes (*Digital_Certificate* and *Certification_Authority*), while the rest of classes are blurred, in order to implement a Focus+Context, which remains the global context and highlights the focused elements.

The names of the coupled classes and the relationships are listed using a circular list that rotates to display the next group of elements. This interaction lets the user to navigate over all the values, without being important the total number of elements.

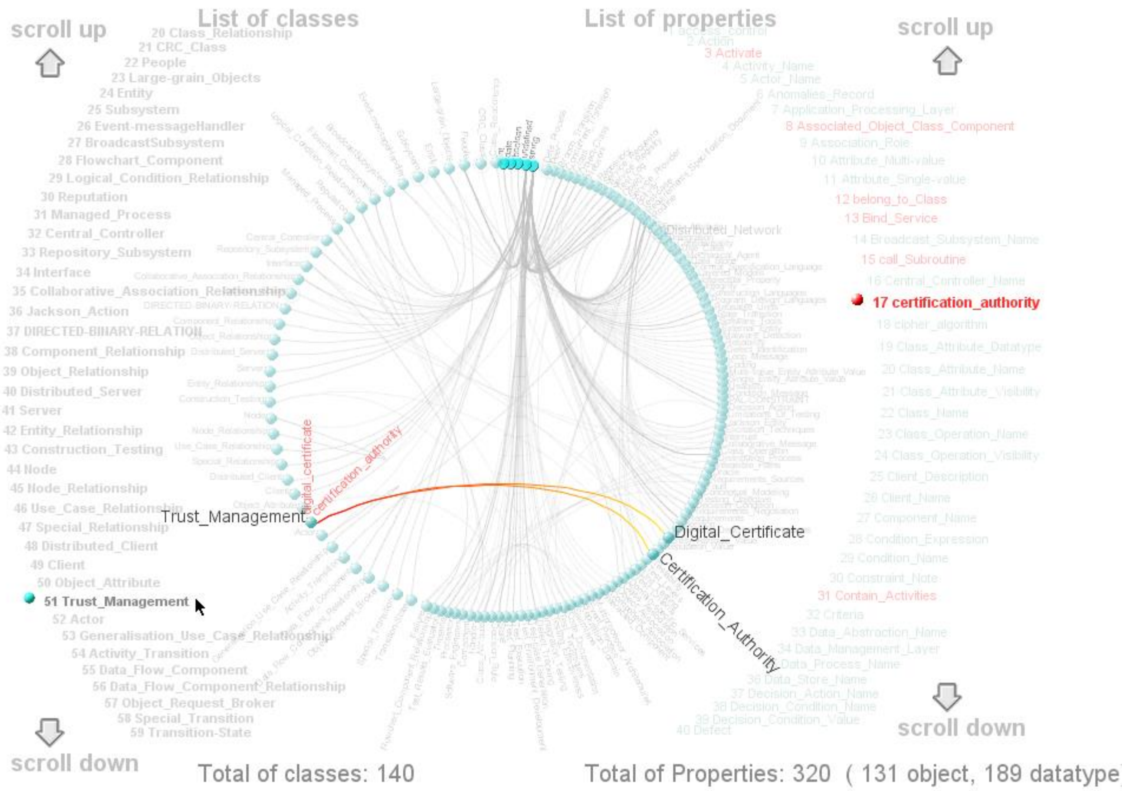


Figure 8 The global coupling visualization shows the relations among classes in the ontology. The class "Trust_Management" has been selected as well as its coupled classes "Digital_Certificate" and "Certification_Authority".

Figure 8 has shown the global coupling of the whole ontology, where all the coupled classes are shown in a global context. Moreover, the user can be interested in the coupling of a specific class, such as the classes *Digital_Certificate* and *Trust_Management*, depicted on Figures 9 (a) and 9 (b). These classes are related or coupled, by means of the relation called *digital_certificate*.

This coupling relationship has been shown on Figure 8, using a general view. The interaction with the user includes the semantic zoom visualization shown on Figures 9 (a) y 9 (b), in response of a user selection of the class. Figure 8 shows this relationship in the sense going from the class *Trust_Management* (in the domain) to the class *Digital_Certificate* (in the range). This direction is indicated with the color of the edge, from the red to the yellow. Moreover, this relation is also depicted on Figures 9 (a) and 9 (b), where the class *Trust_Management* remains to the left side of the class *Digital_Certificate*, indicating the manner that the relation has to be interpreted.

These Figures represent three different views or perspectives of a coupling relationship. The first (Figure 8) is a general view, the second (Figure 9(a)) is a semantic zoom view (detailed view) having as the selected element the class *Digital_Certificate*, while the third one (Figure 9(b)) is a semantic zoom but having as the selected element the class *Trust_Management*.

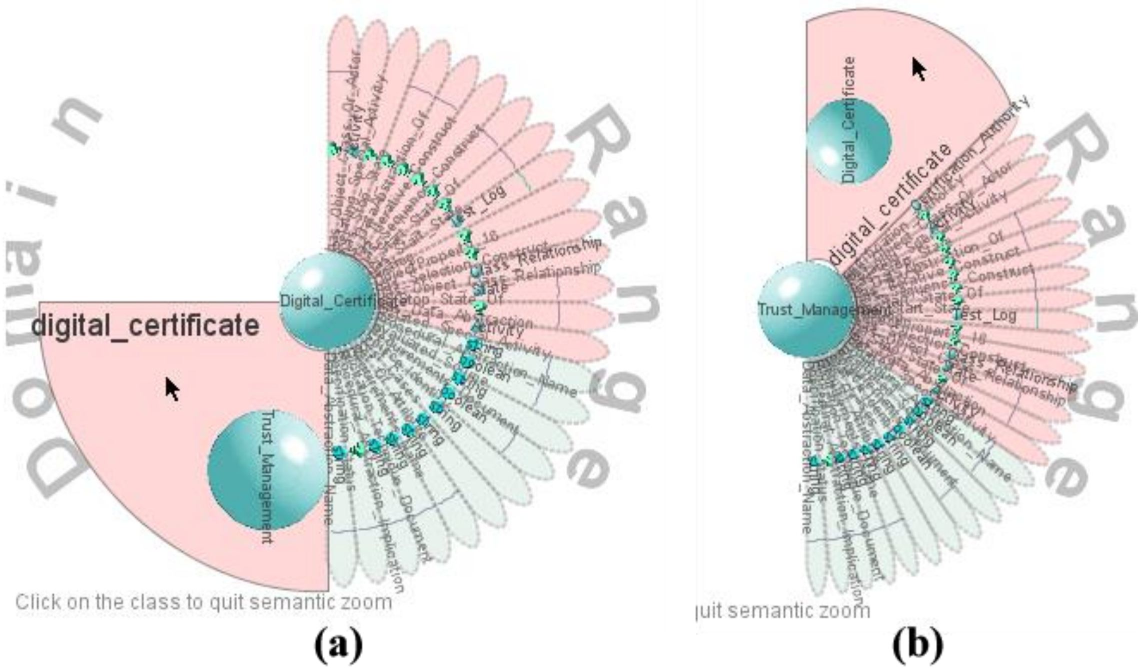


Figure 9 (a) the coupling of the class "Digital_Certificate". Figure (b) shows the coupling of the class "Trust_Management". Both classes are coupled by means of the property "digital_certificate", which indicates that a trusty relationship is performed by means of a Digital Certificate.

Regarding external validity, there are two possible threats. The first is the small number of respondents, which makes difficult the generalization of results. The second is the fact that the sample was not taken randomly. Future works will tackle both threats.

Results & Discussion

Table 1 presents average and standard deviation of the responses offered by the subjects in relation to the questionnaire applied and the two groups of questions formulated.

Table 1. Evaluation of Statistical Results

	OWL-VisMod		Protégé	
	Average	Std. Dev.	Average	Std. Dev.
Taxonomic visualization	7.19	1.123	7.14	1.108
Visual analytics process	6.33	.577	6.48	.512
Ontology modeling workflow	7.38	.805	6.43	1.121
User experience	7.05	.669	6.38	.973
Global performance	6.86	1.108	7.00	.894

Results show that, in general, punctuations are similar for both tools in every aspect. However, it is important to underline that in two factors, namely “Ontology modeling workflow” and “User experience” average values are quite higher in the case of OWL-VisMod, presenting also in these cases lesser and moderate Standard Deviation values also.

In an attempt to verify whether results presented statistically significant differences, the statistical t-test (comparison of two means) was used to analyze if differences between the two groups existed. The level of statistical significance was set at 0.05. The analysis was conducted for each factor. This analysis indicated that two variables present significant differences: User Experience ($t(42) = 2.586$, $p < .05$) and Ontology modeling workflow ($t(42) = 3.162$, $p < .05$). In contrast none of the variables that obtained higher values in Protégé than in OWL-VisMod present significant differences: Global performance ($t(42) = -.460$, $p > .05$) and Visual analytics process ($t(42) = -.849$, $p > .05$). Finally, the variable Taxonomic visualization that obtained higher values for OWL-VisMod does not present significant differences ($t(42) = .891$, $p > .05$).

Results mean that the implementation of OWL-VisMod may be considered a notable success. This tool is comparable in terms of global performance to a standard like Protégé and can provide good results in aspects like User Experience and Ontology Modelling Workflow.

Conclusions

The development of ontologies represents a crucial aspect in the Knowledge Engineering field. Developing an ontological model is very complicated task, independently if the model is built up from the scratch or reusing an existing ontology.

The ontology life-cycle management has all the related issues about knowledge abstraction and modeling with the different notation and methods supported by ontology tools. All these processes present a big gap between the conceptual models in the experts' brains and the ontological models.

OWL-VisMod tool contributes to the development of Ontological Engineering that exploits the formal principles to build ontologies and is based on Visual Analytics techniques to reduce this conceptual gap and close de cycle between the conceptual abstractions and models, allowing the interaction with visual models to extract new knowledge or a better understanding the models in order to iterate in the ontology life cycle processes.

In this paper, OWL-VisMod has been validated performing a set of conceptual modeling tasks using the SEOntology Software engineering ontology. The tasks have been performed using OWL-VisMod and also with Protégé, with the final aim of comparing the results in order to set the validity of OWL-VisMod compared with a recognized tool like Protégé is in the Knowledge Engineering field

Evaluation results show that both tools have similar scores in every aspect, but OWL-VisMod presents better average values in the "Ontology modeling workflow" and "User experience", which it means the importance of a visual analytics in the human depending tasks of knowledge engineering processes.

References

Andrienko, G., Andrienko, N., Jankowsk, P., Keim, D., Kraak, M., MacEachren, A., & Wrobel, S. (2007). Geovisual analytics for spatial decision support: Setting the research agenda. International Journal of Geography Information Science, 21 (8), 839–857.

Baehrecke, E., Dang, N., Babaria, K., & Shneiderman, B. (2004). Visualization and analysis of microarray and gene ontology data with treemaps. BMC Bioinformatics, 5(1), 12–14.

Bederson, B., Shneiderman, B., & Wattenberg M. (2002). Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. AcM Transactions on Graphics, 21(4), 833–854.

Blanco, C., Lasheras, J., Fernández-Medina, E., Valencia-García, R., & Toval, A. (2011). Basis for an integrated security ontology according to a systematic review of existing proposals. Computer Standards & Interfaces, 33(4), 372-388.

Breslin, J.G., O'Sullivan, D., Passant, A., & Vasiliu, L. (2010). Semantic Web computing in industry. Computers in Industry, 61(8), 729-741.

Chicaiza, J., López, J., Piedra, N., Martínez, O., & Tovar, E. (2010). Usage of social and semantic web technologies to design a searching architecture for software requirement artifacts. IET Software, 4(6), 407-417.

Fensel, D. (2002). Ontologies: A silver bullet for knowledge management and electronic commerce. Berlin, Germany: Springer-Verlag.

Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997). Methontology: From ontological art towards ontological engineering. Proceedings of the Ontological Engineering AAAI-97 Spring Symposium Series, 1, 33–40.

Gallardo, J., Molina, A.I., Bravo, C., Redondo, M.A., & Collazos, C.A. (2011). An ontological conceptualization approach for awareness in domain-independent collaborative modeling systems: Application to a model-driven development method. Expert Systems with Applications, 38(2), 1099-1118.

García, J., García-Peñalvo, F. & Therón, R. (2010a), Defining coupling metrics among classes in an OWL ontology, In Trends in Applied Intelligent Systems. 23rd International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, IEA/AIE 2010, Springer LNAI 6097, 12–17.

García, J., García-Peñalvo, F. & Therón, R. (2010b), Visualising semantic coupling among entities in an owl ontology, Proceedings of ONTOSE 2010, Ontology, Conceptualization and Epistemology for Information Systems, Software Engineering and Service Science, Springer 62 (3), 90–106.

García, J., García-Peñalvo, F. & Therón, R. (2011), Modelling relationships among classes as semantic coupling in owl ontologies, Proceedings of the 2011 International Conference on Information & Knowledge Engineering, IKE 2011, 1, 22–28.

García, J., García-Peñalvo, F., Therón, R. & Ordóñez de Pablos, P. (2011), Usability evaluation of a visual modelling tool for owl ontologies. Journal of Universal Computer Science, 17 (9), 1299–1313.

García, J., Therón, R. & García-Peñalvo, F. (2011), Semantic zoom: A details on demand visualisation technique for modelling owl ontologies, Highlights in Practical Applications of Agents and Multiagent Systems, 9th International Conference on Practical Applications of Agents and Multiagent Systems AISC 89, 85–92.

García-Peñalvo, F., García, J. & Therón, R. (2011), Analysis of the owl ontologies: A survey, Scientific Research and Essays, 6(20), 4318–4329.

García, R., Gil, R., Gimeno, J.M., Granollers, T., López, J.M., Oliva, M., & Pascual, A. (2010). A Semantic Wiki for Quality Management in Software Development Projects. IET Software, 4(6), 386–395.

García-Crespo, A., Colomo-Palacios, R., Gómez-Berbís, J.M., & Mencke, M. (2009). BMR: Benchmarking Metrics Recommender for Personnel issues in Software Development Projects. International Journal of Computational Intelligence Systems, 2(3), 257-267.

Gennari, J., Musen, M., Fergerson, R., Grosso, W., Crubezy, M., Eriksson, H., Noy, N., & Tu, S. (2003). The evolution of protege: An environment for knowledge-based systems development. Stanford Medical Informatics, 58(1), 89–123.

Herman, I., Melancon, G., & Marshall, S. (2000). Graph visualization and navigation in information visualisation: a survey. IEEE Transactions on Visualisation and Computer Graphics, 6(1), 24–43.

Holten, D. (2006). Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. IEEE Transactions on visualization and computer graphics, 12(5), 741–748.

Gómez-Pérez, A., Fernández, M., & Corcho, O. (2003). Ontological Engineering. Springer-Verlag.

Haase, P., Lewen, H., Studer, R., & Erdmann, M. (2008). The neon ontology engineering toolkit. In Proceedings of the 17th International World Wide Web Conference.

Hitzler, P., & Janowicz, K. (2011). Semantic Web tools and system. Semantic Web, 2(1), 1-2.

Isenberg, P., & Fisher, D. (2009). Collaborative brushing and linking for co-located visual analytics of document collections. Computer Graphics Forum, 28(3), 1031-1038.

Janev, V., & Vranes, S. (2011). Applicability assessment of Semantic Web technologies. Information Processing & Management, 47(4), 507-517.

Jarrar, M., & Meersman, R. (2002). Formal ontology engineering in the dogma approach. Proceedings of the International Conference on Ontologies, Databases and Applications of Semantics, 2519, 1238–1254.

Jarrar, M., & Meersman, R. (2008). Ontology engineering -the dogma approach. In T.S. Dillon et al. (Eds.) Advances in Web Semantics I, 4891,7–34. Springer Verlag: Berlin.

Jiang, X., & Tan, A.H. (2009). Learning and inferencing in user ontology for personalized semantic web search. Information Sciences: an International Journal, 179(16), 2794–2808.

Johnson, B., & Shneiderman, B. (1991). Treemaps: a space filling approach to the visualization of hierarchical information structures. 2nd. IEEE Visualization Conference, 284-291.

Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., & Giannopoulou, E. (2007). Ontology visualization methods-a survey. ACM Computing Surveys, 39(4).

Kozaki, K., Kitamura, Y., & Mizoguchi R. (2005). Developing ontology-based applications using HOZO. Proceedings of the Fourth IASTED International Conference on Computational Intelligence, 1, 10–16.

Martinho, R., Varajao, J., & Domingos, D. (2010). Using the semantic web to define a language for modelling controlled flexibility in software processes. IET Software, 4(6), 396–406.

Mizoguchi, R. (2004). Tutorial on ontological engineering: Advanced course of ontological engineering. New Generation Computing, 22(2), 198–220.

Saffer, J.D., Burnett, V.L., Chen, G., & van der Spek, P. (2004). Visual analytics in the pharmaceutical industry. IEEE Computer Graphics and Applications, 24(5), 10–15.

Shiva, S.G., & Shala, L.A. (2008). Using Semantic Wikis to Support Software Reuse. Journal of Software, 3(4), 1-8.

Shneiderman, B. (2008). Extreme visualization: squeezing a billion records into a million pixels. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, 3–12.

Sicilia, J.J., Sicilia, M.A., Sánchez-Alonso, S., García-Barriocanal, E., & Pontikaki, M. (2009). Knowledge representation issues in ontology-based clinical Knowledge Management systems. International Journal of Technology Management, 47(1-3), 191-206

Soto-Acosta, P., Casado-Lumbreras, C., & Cabezas-Isla, F. (2010). Shaping human capital in software development teams: the case of mentoring enabled by semantics. IET Software, 4(6), 445–452.

Storey, M.A., Musen, M., Silva, J., Best, C., Ernst, N., Fergerson, R., & Noy, N. (2001). Jambalaya: interactive visualization to enhance ontology authoring and knowledge acquisition in Protégé. In Workshop on Interactive Tools for Knowledge Capture (K-CAP-2001), Victoria, British Columbia, Canada.

Suárez-Figueroa, M., Brockmans, S., Gangemi, A., Gómez-Pérez, A., Lehmann, J., & Lewen, H. (2007). Neon modeling components. NeOn deliverable d5.1.1, Neon Project. Technical report, NeOn project.

Sunagawa, E., Kozaki, K., Kitamura, Y., & Mizoguchi, R. (2005). A framework for organizing role concepts in ontology development tool: Hozo. In Proceedings of the AAAI Fall Symposium Technical Report FS-05-08, Arlington, Virginia, USA, 136–143.

Sure Y., Angele J., & Staab S. (2002). Ontoedit: Guiding ontology development by methodology and inferencing. In Proceedings of On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE, Lecture Notes in Computer Science, Volume 2519/2002, 1205-1222.

Suresh, K., Kumar, S., Prakash, N., & Rizvi, S. (2008). Role of ontology editors: Ontology design. Proceedings of IPCV08.

Tacla, C.A., Freddo, A.R., Paraiso, E.C., Ramos, M.P., & Sato, G.Y. (2011). Supporting small teams in cooperatively building application domain models. Expert Systems with Applications, 38 (2), 1160-1170.

Tappolet, J., Kiefer, C., & Bernstein, A. (2010). Semantic web enabled software analysis. Web Semantics: Science, Services and Agents on the World Wide Web, 8(2-3), 225-240.

Telea, A., & Voinea, L. (2009). Case study: Visual analytics in software product assessments. In Proceedings of the 5th IEEE International Workshop on Visualizing Software for Understanding and Analysis, 2009. VISSOFT 2009, 65–72.

Thomas, J.J., & Cook, K.A. (2005). Illuminating the Path: The Research and Development Agenda for Visual Analytics. IEEE Press: Los Alamitos, California.

Tominski, C., Schulze-Wollgast, P., & Schumann, H. (2008). Visual methods for analyzing human health data. In N. Wickramasinghe & E. Geisler (eds.) Encyclopedia of Healthcare Information Systems, 1, 1357–1364. IGI-Global: Hershey, PA.

Tominski, C., Abello, J., Van Ham, F., & Schumann, H. (2006). Fisheye tree views and lenses for graph visualization. In Proceedings of the International Conference on Information Visualization, 17–24.

Uschold, M., & King M. (1995). Towards a methodology for building ontologies. Proceedings of Workshop on Basic Ontological Issues in Knowledge Sharing, 1, 1–15.

Villalón-Terrazas, B., Ramínez, J., Suárez-Figueroa, M.C., & Gómez-Pérez, A. (2011). A Network of Ontology Networks for building e-Employment Advanced Systems. Expert Systems with Applications, 38(11), 13612-13624.

Ware, C. (2004). Information Visualization: perception for design. Morgan Kaufmann: San Francisco, California.