USABILITY EVALUATION OF ONTOLOGY EDITORS

E. García-Barriocanal, M. A. Sicilia, S. Sánchez-Alonso

Computer Science Department, Polytechnic School. University of Alcalá

Ctra. Barcelona km. 33.6. 28871, Alcalá de Henares. Spain.

{elena.garciab, msicilia, salvador.sanchez }@uah.es

Abstract: Ontology editors are software tools that allow the creation and maintenance of ontologies through a graphical user interface. As the *Semantic Web* effort grows, a larger community of users for this kind of tools is expected. New users include people not specifically skilled in the use of ontology formalisms. In consequence, the usability of ontology editors can be viewed as a key adoption precondition for *Semantic Web* technologies. In this paper, the usability evaluation of several representative ontology editors is described. This evaluation is carried out by combining a heuristic pre-assessment and a subsequent user-testing phase. The target population comprises people with no specific ontology-creation skills that have a general knowledge about domain modelling. The problems found point out that, for this kind of users, current editors are adequate for the creation and maintenance of simple ontologies, but also that there is room for improvement, especially in browsing mechanisms, help systems and visualization metaphors.

Keywords: Usability evaluation, Semantic Web, ontologies, ontology editors.

1. Introduction

1.2. The role of ontology editors in the Semantic Web

From an Artificial Intelligence perspective, ontologies can be described as a kind of knowledge representation (Davis, Shrobe and Szolovits 1993) for shared conceptualizations of specific domains (Decker et al. 2000), which is considered as a key enabling technology for e-commerce (Fensel 2001) and for the so-called Semantic Web (Ding et al. 2002). To date, widely used object-oriented modelling languages like the UML –Unified Modelling Language– (Object Management Group 2003) have been used to represent ontologies (Cranefield and Purvis 1999; Cranefield, Haustein and Purvis 2001). However, current ontology formalisms, like KIF (NCITS 1998), either exceed the built-in information representation capabilities of the core meta-models of those languages (Cranefield and Purvis 1999), or make it necessary the introduction of a set of supplementary notational extensions (Baclawski et al. 2001), both resulting in harder-to-learn modelling languages. In consequence, it is expected that knowledge representation (KR) specific tools, like Protégé (Noy et al. 2001), will continue to be used for ontology creation and editing in the near future, taking into account that frame-based mark-up languages –such as RDF/RDFS and its extensions– are intended for computer interchange rather than for direct human reading and writing..

In this work, the term Ontology editor (OE) is used to refer KR-specific software tools, explicitly based on any ontology formalism, which allow the interactive creation and updating of ontologies through a graphical user interface. The focus of this research is on the specific human-interaction characteristics of these tools, assuming that efficient and easy to use ontology creation and maintenance applications are a critical element in the necessary Semantic Web infrastructure. Taking into account that a larger community of users would include a larger number of non-KR specialists, this study aims at investigating whether current OEs are usable for people without a deep understanding (or experience) in ontology modelling.

The rest of this paper is structured as follows. In the rest of this section the general principles and methods of the evaluation are described. In Section 2, the test procedure is explained in detail, including the findings obtained from the pre-assessment heuristic evaluation. Later on, section 3 includes the results. Finally, conclusions and future research directions are sketched in Section 4.

1.2. Overall description of the evaluation

Usability evaluation is considered an important dimension in the evaluation of systems that have some kind of knowledge acquisition interfaces (Adelman and Riedel 1997). Reports on usability evaluation of various knowledge representation systems have addressed different usability measures, e.g. time to learn specific knowledge entry functionalities (Shahar et al. 1999), as well as technical aspects that directly affect the user: explanation, error handling, system's efficiency and adequacy of programming interfaces (McGuinness and Patel-Schneider 1998).

In previous studies (Duineveld et al. 2000), a comparison between six ontology-engineering tools was made in accordance with three different dimensions: the user interface, the ontology-related issues found in the tool, and the tool's capacity to support the construction of an ontology by several people at different locations. In Duineveld's report (Duineveld et al. 2000), the authors describe their opinion about the ontology engineering tools by using a checklist, but potential actual users did not take part in the evaluation.

In this work, a conventional usability evaluation has been carried out combining two widespread techniques: heuristic evaluation and user testing. Three groups of users/evaluators have been formed, each with different backgrounds, to report on the usability of selected OEs. The main objective is not to analyse specific knowledge entry techniques, but to consider general user interaction issues. An explicit distinction between evaluation and assessment of knowledge sharing technology (KST), which include ontology editors: "Evaluation means to judge technically the features of KST, and assessment refers to the usability and utility of KST in companies" (Gómez-Pérez 1994), has been proposed elsewhere. However, the term evaluation has been kept for the sake of clearness in the application of the most common methods and techniques to measure system's usability, as this is a more familiar term in the Human-Computer Interaction (HCI) community.

Preceded by a heuristic evaluation (Nielsen 1994) aimed at obtaining the present assessment of usability problems, conventional user testing techniques (Dumas and Redish 1999) have been selected as the main approach. These methods are considered complementary, as each one detects usability problems overlooked by the other (Nielsen 1994). Our evaluation is mainly formative, in the sense that it is targeted to expose usability problems in current tools. However, because of the process, some aspects that could be used as the point of departure of a summative evaluation –i.e. one that tries to determine which among several alternatives to ontology editing is best–, have also been identified. In addition, as ontology editors are far too complex to test all their functionalities at a time, the study has been purposefully limited to answering questions relevant to the fact that the community of OE users will grow with professionals that do not come from the KR field.

The general concern of our study is to determine the ease of use of OEs or, in other words, to be able to provide an answer to the question: "if users have limited or null experience in ontology creation and maintenance, or if they have an exploratory learning style, are ontology editors good in terms of usability?" For this purpose, users will be considered to have an exploratory learning style if they prefer to learn about the use of the system by investigating it on their own initiative –often in pursuit of a real or artificial task– instead of working through precisely sequenced training materials. More specifically, the following two concerns have been raised: "How easy is it to create a new ontology with current OEs?" and "how easy is it to browse, search and perform updating tasks on large ontologies with current OEs?". Related activities allowed in some OEs like Protégé (Noy, Fergerson and Musen 2000), such as semantic Web page annotation, collaborative ontology edition, or ontology meta-modelling, are not considered here, since they are not directly supported by the most commonly used OEs.

The following tools were initially selected for inclusion in the test: [a] Protégé 2000 1.6.2 [1] [b] OntoEdit 2.0 [2], [c] OILEd 2.2a [3], [d] the KSL Ontology Editor[4] [e] WebOde 1.1 [5], [f] WebOnto[6], and [g] KADS22 [7] . Although several OE currently exists —an exhaustive list can be found in [Denny, 2002]— , the stability of the version as well as platform and licensing constraints have served as a filter in the selection of editors made for this study. From the selected OEs, those that do not allow both edition and creation processes were discarded. Later, practitioners were asked for their opinion on the most widely used, obtaining the enumeration above. In the overall process of selection, the main criteria was that of comparing two groups of interfaces: HTML-based interfaces and 'GUI-desktop' interfaces. In the rest of this paper, the editors listed will be referenced by the letter showed in brackets. .

2. Evaluation design

2.1. Specific concerns and measures

The specific concerns of the evaluation were motivated by a heuristic analysis (Nielsen 1994) carried out by experts with at least one year of previous experience in ontology edition. The procedure for the evaluation consisted on three phases: a pre-selection phase, in which some tools could be discarded, the actual evaluation, and a debriefing and severity-rating phase. Although experts were free to take their own approach, they were suggested to edit simple ontologies taken from the Internet, to browse sample ontologies downloaded from the DAML library

---

[1] Features, functionalities and downloads from http://protege.stanford.edu/

[2] Features, functionalities and downloads from http://www.ontoprise.de

[3] Features, functionalities and downloads from http://oiled.man.ac.uk

[4] Features, functionalities and downloads from http://www-ksl-svc.stanford.edu:5915/

[5] Features, functionalities and downloads from http://kw.dia.fi.upm.es/wpbs/

[6] Features, functionalities and downloads from http://kmi.open.ac.uk/projects/webonto/

and to search in the $(KA)^2$ ontology (Benjamins et al. 1999); the latter two tasks, were only performed for those OEs including support for loading RDF ontologies. As four evaluators carried out the study, more than fifty percent of the usability problems are estimated to be found according to Nielsen's curve in (Nielsen 1992).

In the first phase –after the first three experts' pre-evaluations– KADS22 was discarded. This decision was based on the fact that it does not adhere to common platform conventions, as well as its clear orientation to CML-file editing (it is important to note that the tool considers itself as to be 'in development phase'). In addition, WebOde was not evaluated, since it mixes HTML forms with graphical interfaces based on applets, which can difficult the categorization and comparison to the two target groups of tools. Table 1 summarizes the most relevant results of the second phase of the analysis, structured around Nielsen's heuristics –the experts were suggested to use Tognazzini's principles (Tognazzini 2002) as a checklist–. The column marked 'S', shows the *severity* estimated by the experts in the third phase. According to Nielsen (Nielsen 1994), a scale from 0 (no problems) to 4 is used, were 1 stands for *cosmetic problems*, 2 for *minor problems*, 3 for *major problems* and 4 for problems that are *imperative to be fixed*. When an expert detects a problem on an editor, this should be recorded by marking an 'x' in the corresponding column. The acronym *n.a.* stands for *not applicable*.

The main conclusion of the heuristic evaluation is that major usability problems are scarce, except from the lack of appropriate help and user error reporting systems.

After the heuristic evaluation, WebOnto was discarded from the user test due to several behaviour problems related to the interface (buttons disappearing in the toolbar, operations that do not report errors but do nothing, and the like). This incorrect behaviour was possibly due to minor issues, such as a non-compatible Web browser version or any other problem related to the common platform, but it made it impossible to carry out a fair comparison with the other tools. Nonetheless, the graphical editing capabilities of WebOnto, that provide an appropriate and efficient way to edit hierarchies, as well as the unique collaborative edition capabilities, must be highlighted.

Heuristic	Problems Found	S(0..4	A	B	C	D	F
Visibility of system status	Lack of status bar	2	X	-	-	X	X

[7] Features, functionalities and downloads from http://hcs.science.uva.nl/projects/kads22/

Match between system and the real world	Unexplained system-oriented terms	3	-	-	X	-	X
	No printing functionality	3	X	X	X	X	X
User Control and freedom	No "un-do"/"re-do" functionality	3	X	-	X	.	X
	No "replace" functionality	2	X	X	-	X	X
	No "copy & paste" functionality in hierarchies	2	X	-	X	X	X
	No "drag & drop" functionality in hierarchies	2	-	X	X	X	X
	No "cut & paste" functionality in hierarchies	2	X	-	-	X	X
	No tool tips in some elements	2	X	-	-	X	X
	Pop up menu navigation using cursors is not	1	X	X	-	na	na
	No searching slots functionality	2	X	X	-	-	-
Consistency and standards	Does not follow menu platform conventions	1	-	-	-	-	X
Recognition rather than recall	Actions available only through	2	-	-	X	na	na
Flexibility and efficiency of use	Excessive time to launch	3	-	-	-	X	X
	No key accelerator	2	X	X	-	na	X

Table 1. Heuristic evaluation results.

Using the heuristic evaluation results as a basis for discussion, the general concerns of the test are detailed in what follows in specific issues, and the measures used for each of these issues are provided. First of all, the general concern addressing the question *"How easy is to create a new ontology with current OEs?"* has been detailed in two issues:

a) How easy is it to create a new empty ontology and setting the initial basic properties?

b) How easy do new users find the definition of a new ontology construct of type X (where X stands for, respectively, a class, a property and instance)?

Next, the specific issues derived from the second general concern, *"how easy is it to browse, search and perform updating tasks on large ontologies with current OEs?"*, are the following:

a) How easy is to find a specific ontology construct of type X?

b) How easy is it to navigate through the generalization/specialization hierarchy?

c) How easy is it to update a characteristic C (e.g. name, property/slot, instance) of an existing ontology construct of type X?

In all the cases, the time to complete the task and the number of errors raised in it have been selected as a measure for the issue. The application response time has not been included in the evaluation, as it is easy to check that some of the current OEs would require further improvements in parsing and/or caching of large ontologies. An example is the large *Universal Standard Products and Services Classification* ontology that takes about two minutes to load in Protégé 2000 on a Pentium III computer with 1GB of main memory. Loading this particular ontology makes the Protégé process to grow up to 150 MB of memory size. In addition, Web-based OEs, in some cases, do not reach the 1 second-limit necessary to keep uninterrupted the user's flow of thought (Nielsen, 2000), although do not violate the 10 second response time limit that is considered for keeping the user's attention focused on the dialogue.

2.2. Participants

The target population is composed of individuals who share the following characteristics: more than five years of experience in the use of computers, daily use of complex GUI-based applications, and a minimal understanding of conceptual models (but capable of understanding, at least, simple UML class diagrams). A pre-test phase allowed discarding users not fitting this profile, as current OEs are not considered adequate for them. Note that the ontologies used in the test are designed for usability rather than for reusability –in the sense given in (Domingue and Motta 1999)–, and therefore, further testing would be required in ontologies designed for reusability. Moreover, a number of features that can be considered as *advanced* ontology modelling, like exploiting inference engines or defining axioms through formulas, were left apart from the analysis. An informal experiment with three not knowledgeable in KR Internet services users pointed out that including those features is simply not realistic. The experiment consisted in editing axioms with OILEd from natural language descriptions. None of the users was able to complete the task in a reasonable time, which suggests that simpler and more intuitive interface metaphors are required for those tasks to be carried out by people with no background in description logics or similar formalisms. From that basic user profile, three subgroups were considered: (1) Users with experience in ontology definition, (2) users with experience in computer-based modelling (e.g. users with experience in UML modelling tools) but with no experience in ontology definition and (3) users with neither experience in computer-based modelling nor ontology definition, but accustomed to use computer applications. For the test, four participants in each group were selected. In addition, a participant in each of the subgroups was selected for performing a pre-test oriented towards detecting defects in the test process itself.

2.3. Procedure and scenarios

The final test comprised the three following steps:

1 *Learning step*. Participants in subgroups 2 and 3 were given a brief introduction both to general ontology concepts and to the specifics of every OE under evaluation. For subgroup 2, the explanations were structured around concepts that are not usual to UML users, like, for example, the fact that properties are a first-class modelling element (Baclawski et al. 2001).

2 *Evaluation.* It was divided into two parts, one for each general concern.

3 *Post-test.* After each part of the evaluation, the participants responded to a satisfaction questionnaire aimed at measuring their subjective satisfaction.

In order to evaluate the specific issues detailed in section 2.1, scenarios in step two were set up as follows:

- *Scenario 1.* The user creates a small ontology from scratch. For this purpose, a part of the ontology described in (Fensel et al. 2000), written in a language neutral (from the perspective of ontology languages) textual form, and sketched as a UML diagram (a total of ten classes, and five properties), was used. This scenario was the same for all the evaluations.
- *Scenario 2.* After loading a (relatively) large ontology, the user is requested to search a class and a property, to annotate all the relationships of the class (along with the entire generalization hierarchy), and to perform small updates on either the class, or the property, or both. Depending on the editor, different ontologies were used: *Cyc-Transportation Ontology*, *World-Fact-Book* and *UNSPSC*.

2.4. Tools and environment of the test

The test team was made up of three of the experts involved in the heuristic evaluation phase. The environment in which the test was carried out was an isolated room with a personal computer running the Windows 2000 operating system in it. The user interaction was recorded with screen capture software while one of the experts observed the participant, so that the expert could focus on observing the reactions of the user. Each participant evaluated all the OE, but the order of evaluation was different for each participant in order to prevent biases derived from remembering previous scenarios.

3. Results

3.1. Test results

Figure 1 summarizes the overall results obtained for each editor. They were obtained by calculating the arithmetic mean of the time, in minutes, that each group (represented by the arithmetic mean of the minutes of its members) uses to complete the first and second scenarios, respectively. Figure 2 shows more in detail, for each editor, the time (in seconds) spent by the groups in performing a task, and the number of errors they made before the task was completed. Both measures are the arithmetic mean of the members of the group.

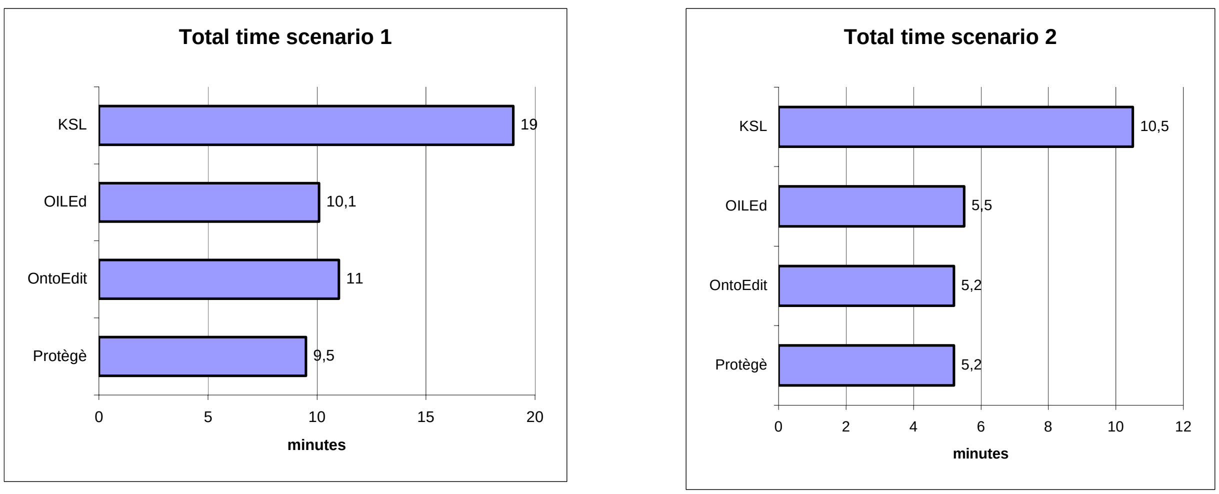
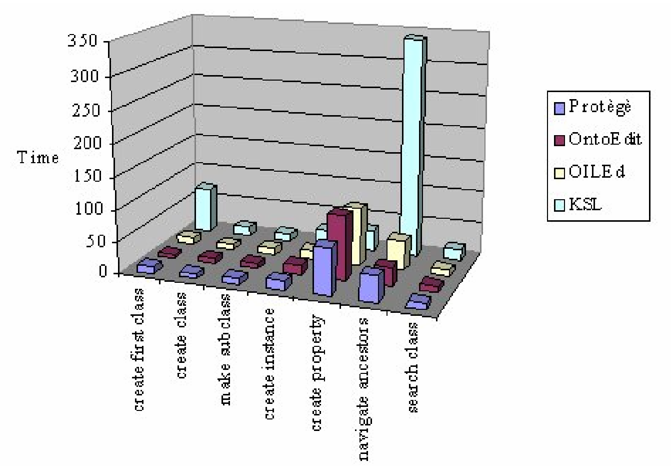
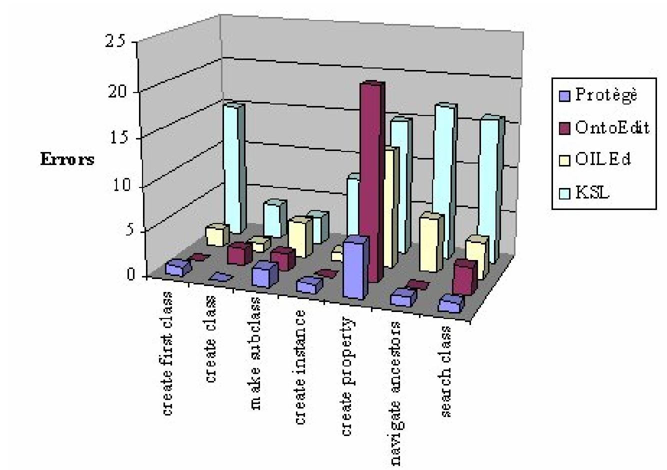


Figure 1: Overall results classified by ontology editor

In some cases, as for example the 'create class' task, the scenario involved several repetitions, and thus the time is the average time to complete. The measures should be considered approximate, since most of the users did not take a task-by-task approach, but instead they explored the interface options, performing partial tasks that were completed later.

The KSL ontology editor exhibited problems in both orientation and navigability (e.g. frames that hided some functions, errors that did not provide links to go back, and users that found it difficult to know what they were editing). All this is possibly the cause for the significantly higher times and error rates, which also increased significantly in the third group. Some participants in this group were not able to complete the tasks in the estimated maximum time. In addition, the pages of the KSL editor do not fit the common visualization area of a browser, what results in scrolling and frame resizing; what significantly increases time-to-complete. These specific problems prevented to make a fair comparison between HTML-based and desktop-based interfaces.





	create first class	create class	make subclass	create instance	create property	navigate ancestors	search class
Protègè	10	8	10	15	74	41	9
OntoEdit	6	8	9	15	100	29	9
OILEd	11	8	10	14	90	47	9
KSL	72	15	11	24	32	340	18

	create first class	create class	make subclass	create instance	create property	navigate ancestors	search class
Protègè	1	0	2	1	6	1	1
OntoEdit	0	2	2	0	21	0	3
OILEd	2	1	4	1	13	6	4
KSL	15	4	3	8	15	17	16

Figure 2: Time (in seconds) and number of errors needed to complete each task in each editor.

OntoEdit measures show that only property-related operations were problematic for users, perhaps because most of them defined properties at a global level, thus preventing the existence of a clear way to attach them to previously defined elements, which caused disorientation. OILEd measures are of a similar magnitude, but specific problems arise in navigating the class hierarchy. Protégé measures are slightly better than those of OntoEdit, but no significant conclusions can be drawn from them. The metamodel accessibility, both in Protégé and in KSL, is perceived as a drawback that causes errors and disorientation, since non-specialists hardly understand the need for such functionality. An overall analysis reveals that browsing large hierarchies can be considered a time-consuming task, and that creating properties is an error-prone activity, perhaps due to the duality between global and local properties.

3.2. Post-questionnaire results

In order to understand usability, it is important to not only measuring user performance (effectiveness and efficiency) but also user satisfaction. A slightly modified version of the *System Usability Scale* (Brooke 1986) was used in a simple, five-item Likert scale (from 1-completely disagree to 5-completely agree) questionnaire. This questionnaire gives a global view of subjective assessments of usability, whose results are summarized in Figure 3. Note that the help system was not evaluated, since it was clearly identified as an improvement area in the heuristic analysis phase.

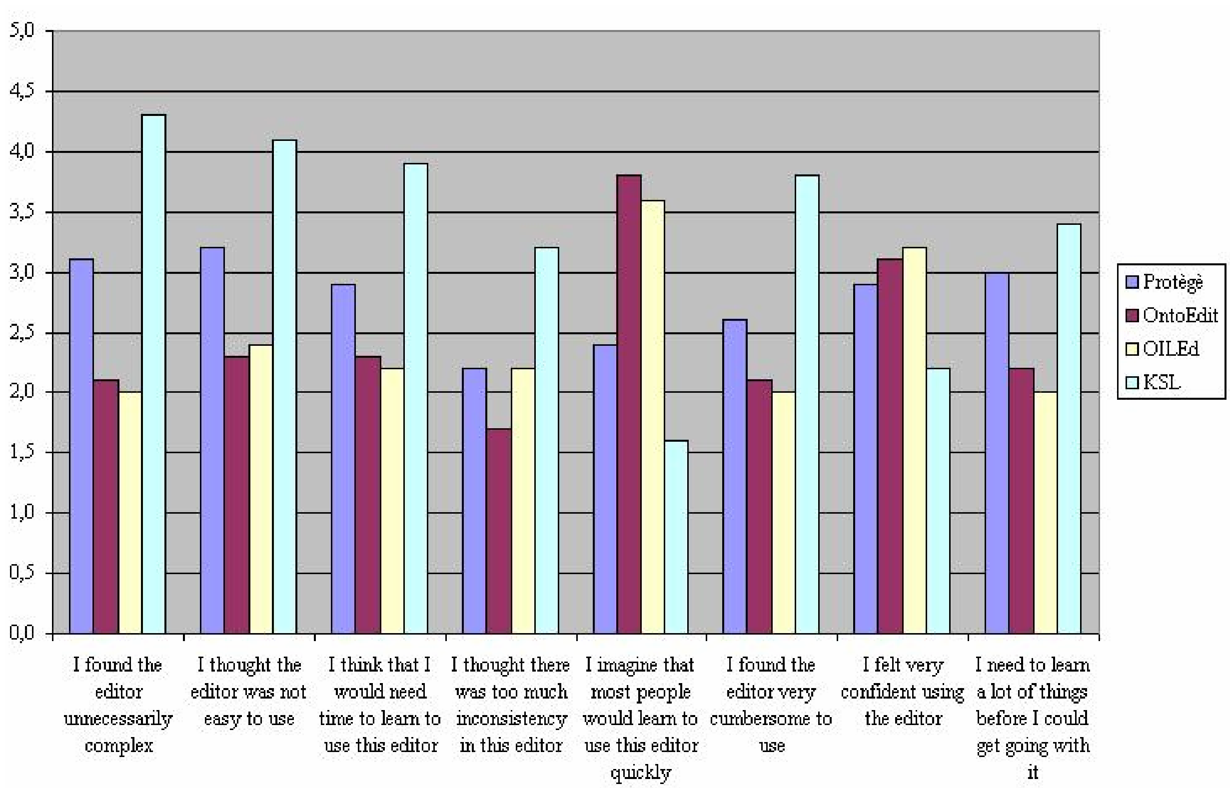


Figure 3: User satisfaction .

The global scores clearly show that the KSL editor is perceived as a complex and difficult to use editor, with a clear difference from the rest. The notes of the evaluators corroborate this fact, as six of the participants complained about KSL, while four of them pointed out that the problem was that HTML-based interfaces are, in general, less usable. The high score in question 1 may respond to the fact that most of the users found themselves *lost* while navigating in the KSL metamodel, because it is accessible through links in the editor. OILEd and OntoEdit obtain similar satisfaction scores, and are significantly perceived as easier than Protégé, except from questions 4 and 7, that are directly linked to predictability. This result points out that some editing capabilities in Protégé that could be considered as *advanced* –such as the explicit edition of the metamodel–, are perceived as unnecessary complex for non-specialists. Another important conclusion is that no relevant differences between the three user profiles exist, apart from a slight increase in the perception of easiness in the third group.

The global satisfaction results from the three desktop editors show that all of them can be considered reasonably adequate for their purposes.

3.3. Summary of major problems

To summarize the study, a list with the most relevant improvement areas detected was elaborated:

1. Integrated, context-aware help systems should be developed.
2. The metamodel should be considered an advanced feature, and thus, it should be disabled by default. In addition, the use of a common metamodel terminology across editors would be beneficial (e.g. providing a unified name to the concept of *relation between classes*, since this is currently referred to as property, relation or slot depending on the editor), in order to hide the differences between the underlying ontology formalisms as much as possible.
3. The language used in the tools should be oriented towards a non-specialized user community, thus avoiding language-specific constructs and terms.
4. New interaction mechanisms to browse the generalization/specialization hierarchy should be explored. In this sense, edition should be based on a hierarchically structured view. Moreover, as in Protégé, hints should be given to recognize multiple inheritance.
5. Richer navigation and filtering mechanisms should be developed, according to the user task model, as for example, the ability to navigate from a class to its instances, or that of filtering the visualization of classes by given criteria.

4. Conclusions

The overall conclusion is that current GUI-desktop-base ontology editors are fairly adequate for new users that prefer exploratory learning. A number of minor usability errors, which could be easily fixed, have been reported in this paper. In addition, a number of overall improvement areas have been identified, which may be the topic of future research work.

As suggested by the evaluations, new visualization metaphors (e.g. 3-dimensional, filters on the class hierarchy) should be explored, since discovering the hierarchy of a specific class has revealed to be a time-consuming task.

A more comprehensive evaluation is needed, in both the number of editors (including *WebOnto* and *WebOde*, which posses interesting user interface characteristics) and the depth of the analysis. During this study, the authors observed (as it was previously supposed) that users usually prefer learning about how to use an ontology editor by directly using the tool, instead of by reading the documentation. For that reason, the cognitive walkthrough technique (Polson et al. 1996), which pays a special attention to how well the interface supports *exploratory learning*, could be an interesting candidate for further evaluations, including this new concern about the interface in those to be studied.

References

Adelman, S.L., Riedel, L. 1997. Handbook for evaluating knowledge-based systems: Conceptual framework and compendium of methods. Boston, USA: Kluwer Academic Publishers.

Brooke, J. 1986. System usability scale (SUS), User information Architecture A/D Group, Digital Equipment Co. Ltd. Online at: http://www.usability.serco.com/trump/

Baclawski, K., Kokar, M.K., Kogut, P.A., Hart, L., Smith, J.E. , Holmes, W.S., Letkowski, J., Aronson, M.L. 2001. Extending UML to support ontology engineering for the Semantic Web. In Martin Gogolla and Cris Kobryn (eds.) UML 2001 - The Unified Modeling Language. Modeling Languages, Concepts, and Tools. 4th International Conference, Toronto, Canada, October 2001, Proceedings. Springer-Verlag LNCS (2185): 342–360.

Benjamins, R. Fensel, D. Decker, S. Gómez-Perez, A. 1999. $(KA)^2$: Building ontologies for the Internet: A mid-term report. International Journal of Human-Computer Studies 51(3): 687-712.

Cranefield, S., Purvis, M. 1999. UML as an ontology modelling language. Proc. of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence. Stockholm, Sweden.

Cranefield, S., Haustein, S., Purvis, M. 2001.UML-based ontology modelling for software agents. Proc. of the Workshop on Ontologies in Agent Systems, 5th International Conference on Autonomous Agents, Montreal, Canada.

Davis, R., Shrobe, H., Szolovits, P. 1993. What is a knowledge representation? In AI Magazine 14(1): 17-33.

Decker, S.D. Fensel, F., Harmelen, I., Horrocks, S., Melnik, M., Klein, J. B. 2000. In Knowledge representation on the Web, Proceedings of the 2000 International Workshop on Description Logics. Aachen, Germany: CEUR Workshop proceedings, online at: http://CEUR-WS.org/Vol-33/

Denny, M. 2004. Ontology Building: A Survey of Editing Tools. In O'Reilly XML.com. Online at: http://www.xml.com/pub/a/2004/07/14/onto.html

Ding, Y., D. Fensel, Klein, M., Omelayenko, B.. 2002. The Semantic Web: Yet another hip?. In Data and Knowledge Engineering 41(2-3): 205–227.

Domingue, J., Motta, E. 1999. A knowledge-based news server supporting ontology-driven story enrichment and knowledge retrieval. In Proceedings of the 11th European Workshop on Knowledge Acquisition, Modelling, and Management (EKAW '99), LNAI 1621, Springer-Verlag.

Duineveld, A. J., Stoter, R., Weiden, M. R., Kenepa, B., Benjamins, V. R. 2000. Wondertools? A comparative study of ontological engineering tools. In International Journal of Human-Computer Studies, 52(6): 1111-1133.

Dumas, J. S., Redish, J.C. 1999. A practical guide for usability testing. Sheffield, UK: Intellect, Ltd.

Fensel, D., Horrocks, I., Van Harmelen, F., Decker, S., Erdmann, M., Klein, M. 2000. OIL in a nutshell. In Proceedings of the European Knowledge Acquisition Conference (EKAW-2000), Dieng et al. (eds.), Lecture Notes in Artificial Intelligence (LNAI), Springer-Verlag.

Fensel, D. 2001. Ontologies: Silver bullet for knowledge management and electronic commerce. Berlin, Germany: Springer-Verlag.

Gómez-Pérez, A. 1994. Some ideas and examples to evaluate ontologies. Tech. Report # KSL-94-65, Knowledge Systems Laboratory. Stanford University. Online at: ftp://ftp.ksl.stanford.edu/pub/KSL_Reports/KSL-94-65.ps.gz

McGuinness, D.L., Patel-Schneider, P.F. 1998. Usability issues in knowledge representation systems. In Proceedings of the 15th National Conference on Artificial Intelligence. Madison, Wisconsin, USA: American Association on Artificial Intelligence: 608-614.

NCITS (National Committee for Information Technology Standards) - TC T2 (Information Interchange and Interpretation). 1998. American national standard for Knowledge Interchange Format Proposed Draft, NCITS.T2/98-004. Online at: http://logic.stanford.edu/kif/dpans.html

Nielsen, J. 1992. Finding usability problems through heuristic evaluation. Proceedings of the Computer-Human Interaction Conference (CHI-92), Monterey, California (USA), ACM: 373-380.

Nielsen, J. 1994. Heuristic evaluation. In Usability Inspection Methods. John Wiley & Sons.

Nielsen, J. 2000. Designing Web usability: The practice of simplicity. Indianapolis, USA: New Riders Press.

Noy, N.F., Fergerson, R.W., Musen, M.A. 2000. The knowledge model of Protege-2000: Combining interoperability and flexibility. In Proceedings of the 2nd International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000), Juan-les-Pins, France.

Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R.W., Musen, M.A. 2001. Creating Semantic Web contents with Protege-2000. In IEEE Intelligent Systems 16(2): 60-71.

Object Management Group. 2003. The Unified Modeling Language specification, version 1.5. Online at: http://www.uml.org

Polson, P.G., Lewis, C., Rieman, J., Wharton, C. 1992. Cognitive walkthroughs: a method for theory-based evaluation of user interfaces. In International Journal of Man-Machine Studies 36(5): 741-773.

Shahar, Y., Chen, H., Stites, D. P., Basso, L., Kaizer, H., Wilson D. M., Musen, M. A. 1999. Semiautomated entry of clinical temporal-abstraction knowledge. In Journal of the American Medical Informatics Association 6 (6): 494-511.

Tognazzini, B. 2002. First principles and design. http://www.asktog.com/basics/firstPrinciples.html