Antonio Fernández-Caballero (Ed.)

Metodologías de Desarrollo de Interfaces de Usuario Dinámicas

Desarrollo de Interfaces de Calidad

I Jornada sobre Metodologías de Desarrollo de Interfaces de Usuario Dinámicas
Albacete, 11 de Junio del 2004
Actas

Publica: Ediciones José Antonio Gallud
Albacete, España

Materias CDU: 378

ISBN: 84-609-3957-X

Actas de la

I Jornada sobre Metodologías de Desarrollo de Interfaces de Usuario Dinámicas [1]

Albacete, 11 de Junio del 2004

Comité Organizador

Presidente: Pascual González
Coordinador: Antonio Fernández-Caballero
Vocales: María Dolores Lozano, José Antonio Gallud

[1] Financiado por el proyecto METODOLOGÍAS DE DESARROLLO DE INTERFACES DE USUARIO DINÁMICAS de la Junta de Castilla-La Mancha y del Fondo Social Europeo, PBC-03-003.

Introducción

El proyecto "Metodologías de Desarrollo de Interfaces de Usuario Dinámicas" es un proyecto coordinado, de tres años de duración, financiado por la Junta de Comunidades de Castilla-La Mancha y el Fondo Social Europeo, con referencia PBC-03-003. El proyecto surge de la iniciativa del grupo de investigación LoUISE (Laboratory of User Interaction and Software Engineering) de la Universidad de Castilla-La Mancha y cuenta con la colaboración de las dos universidades siguientes: Universidad Politécnica de Valencia (DSIC – Departamento de Sistemas Informáticos y Computación) y Universidad Miguel Hernández de Elche (CIO – Centro de Investigación Operativa).

Esta primera edición de esta jornada ha tenido lugar en Albacete, el día 11 de Junio del 2004, organizada por el grupo LoUISE en las instalaciones del Instituto de Investigación en Informática de Albacete (I3A). Esta I Jornada de Metodologías de Desarrollo de Interfaces de Usuario Dinámicas ha servido de marco para el intercambio de ideas, puesta en común de resultados y fomento de nuevas colaboraciones entre los investigadores del proyecto.

Las presentes actas recogen un total de 8 contribuciones con el denominador común de la búsqueda de un desarrollo de interfaces de usuario dinámicas de calidad; de ahí el subtítulo dado al presente volumen: "Desarrollo de Interfaces de Calidad".

Junio 2004 Antonio Fernández-Caballero

Tabla de Contenidos

Usability and Accessibility Oriented Development Process 1
María Dolores Lozano, Francisco Montero, Pascual González (LoUISE, Universidad de Castilla-La Mancha)

Soporte Formal para Entornos Visuales de Modelado 17
Artur Boronat, José Á. Carsí, Isidro Ramos, Julián Pedrós (DSIC, Universidad Politécnica de Valencia)

Usability Metrics in Adaptive Agent-Based e-Learning Systems 31
Víctor López-Jaquero, Antonio Fernández-Caballero, Pascual González (LoUISE, Universidad de Castilla-La Mancha)

A Controlled Experiment for Measuring the Usability of WebApps Using Patterns 37
Francisco J. García, María Lozano, Francisco Montero, José A. Gallud, Carlota Lorenzo (LoUISE, Universidad de Castilla-La Mancha)

Aproximación a la Evaluación de Interfaces de Realidad Virtual 51
Arturo Simón, José Pascual Molina, Pascual González (LoUISE, Universidad de Castilla-La Mancha)

Navegación mediante Web Semántica: Una Nueva Arquitectura 61
Oscar Martínez, Federico Botella (COI, Universidad Miguel Hernández de Elche), Antonio Fernández-Caballero (LoUISE, Universidad de Castilla-La Mancha)

Uso de Técnicas de Data Mining para la Clasificación de Sesiones de Navegación, a partir de los Ficheros de Registro de un Servidor 69
Alejandro Rabasa, Enrique Lazcorreta, Federico Botella (COI, Universidad Miguel Hernández de Elche), Antonio Fernández-Caballero (LoUISE, Universidad de Castilla-La Mancha)

Buscando el Perfil de Usuario a partir de su Navegación en la Web 81
Arturo Peñarrubia, Antonio Fernández-Caballero, José M. Gascueña (LoUISE, Universidad de Castilla-La Mancha)

Usability and Accessibility Oriented Development Process

María Dolores Lozano, Francisco Montero and Pascual González

Laboratory of User Interaction and Software Engineering
Computer Science Research Institute
University of Castilla-La Mancha, Albacete (Spain)
{mlozano, fmontero, pgonzalez}@info-ab.uclm.es

Abstract. Usability has become a critical quality factor of software systems. Although it was defined as one attribute of software quality in the first quality decompositions at the end of the 70's, it has not been really considered and it has been historically relegated in relation to other quality factors such as efficiency, reliability, safety and so one. For this reason, most software development methodologies do not include mechanisms to take into account this critical factor in the software process. Fortunately, nowadays the point of view is changing and not only the original concept of usability is getting more and more important but also new concepts are emerging such as universal usability, accessibility and so on. In this paper we aim to analyze this problem and give a step forward proposing a usability and accessibility-oriented methodological framework which takes into account these criteria from the very beginning of the software development process.

1 Introduction

Universal access has emerged as a new critical quality target due to the wide spread of the new information technologies that are changing the current society. The appearance of these new technologies might be considered as the origin of a new Era, called the Information Society. New Technologies are being incorporated in different social aspects such as e-commerce, e-government, e-learning and thus, the way in which citizens interact with these new services is one the governments' main worries. The most difficult goal is to achieve an Information Society "for all", with an easy access for anyone, anytime and anywhere [Ste01a]. To attain this, it is very important that software developers take these matters into account when designing these new interactive systems.

As Stephanidis states in [Ste01b], in the context of human-computer interaction Universal Access introduces a new perspective that attempts to accommodate a very wide range of human abilities, skills, requirements and preferences in the design of computer-based products. Even more, "access" is not sufficient to ensure successful usage [Shn00]. In this sense a new concept, Universal Usability, has emerged as an important issue and a topic for computing research. Thus, both concepts, Universal Access and Universal Usability, can be considered complementary to achieve the goal of an Information Society for all.

To attain Universal Usability [Shn00] we have to face three challenges:

- Technology Variety. Supporting a broad range of hardware, software and network access.
- User Diversity. Accommodating users with different skills, knowledge, age, gender, disabilities, culture, etc.
- Gaps in User Knowledge. Bridging the gap between what users know and what they need to know.

This is not a new idea, as a matter of fact, from the end of the 70's, usability was already considered one more attribute of software quality, but historically its importance has been relegated in relation to other quality factors such as efficiency, reliability, safety and so one. For this reason, most software development methodologies do not include mechanisms to take into account this critical factor in the software process. Fortunately, nowadays the point of view is changing and not only the original concept of usability is getting more and more important but also new concepts, such as accessibility which help to attain the goal of an Information Society for all, are emerging.

In this paper we aim to analyze this problem and give a step forward proposing a usability and accessibility-oriented methodological framework, which takes into account these criteria from the very beginning of the software development process.

The structure of the paper is the following. In section 2 we introduce the concept of quality, including the definition of quality models. Then in section 3 we analyze the problematic of integrating usability and accessibility into the software development process, we analyze some approaches and discuss their deficiencies. In section 4 we present a usability and accessibility-oriented methodological framework which aims to give a step forward in this field and contribute some solutions to the problem of incorporating usability and accessibility criteria within the software process. We finish in section 5 with some concluding remarks.

2 Quality Factors within the Software Process

The quality of a software system is a property difficult to define and capture in an operational way. Quality may depend on task-related factors, performance-related factors and development-related factors. The ISO 9126 definition of quality for software products is the totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs (Figure 1).

To accurately measure the abstract concept of "quality", quality models are needed.

A quality model [Bra02] specifies which properties are important in a particular environment. It defines which attributes are important to measure software quality, then a weighting of these attributes is established, and finally measurement methods for these attributes are defined. We use quality models to quantify the quality level of a concrete software application.

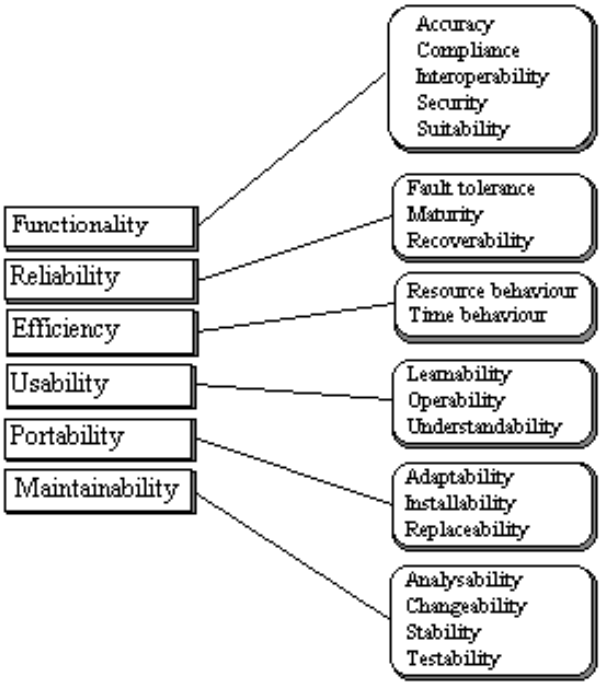


Fig. 1. Software Quality Model (ISO/IEC 9126-1)

The definition of a quality model is based on several steps [IEEE-1061], [Bas94], [Ols02]:

- Establish software-quality requirements, goals and specification. A list of quality factors is selected, prioritized, and quantified from the outset of the system development.
- Identify software-quality metrics.
- Elemental and global evaluation.
- Analyze the software-quality metrics results and validate the software quality metrics.

Considering human-computer interaction, Usability is the main quality attribute to take into account.

Usability is the measure of the quality of a user's experience when interacting with a product or system, whether a Web site, a software application, mobile technology, or any other user-operated device.

ISO/IEC 9126-1 [ISO00] has recently been replaced by a new four part standard that has reconciled the two approaches to usability. ISO/IEC 9126-4 [ISO01] describes the same six categories of software quality: functionality, reliability, usability, efficiency, maintainability and portability. The definition of usability is similar to the previous one:

> *Usability: the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.*

The phrase "when used under specified conditions", is equivalent to the "context of use" concept defined in ISO 9241-11 [ISO98]. This concept was added to make it clear that a product has no intrinsic usability, only a capability to be used in a particular context. In Figure 2, we can see the evolution of the term usability in both ISO international standards. As we can see, both usability definitions are compatible.

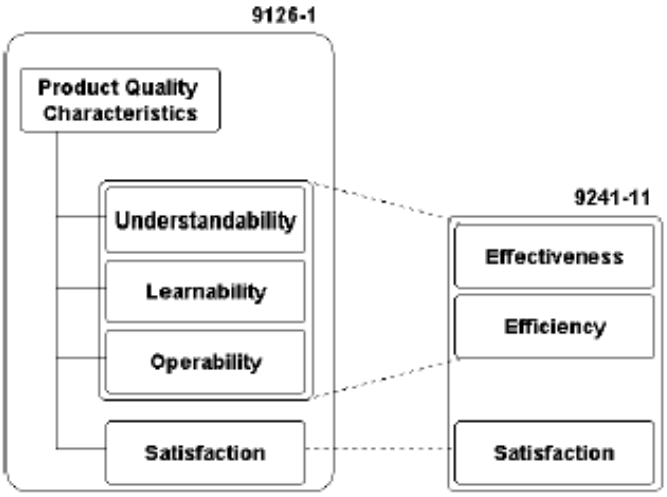


Fig. 2. Usability and International standards

Usability is a combination of factors that affect the user's experience when interacting with a product or system. It includes:

- Understandability: How fast can a user who has never seen the user interface before learn it sufficiently well to accomplish basic tasks?
- Learnability: if a user has used the system before, can he or she remember how to use it effectively the next time or does the user have to start over again learning everything?
- Operability: how fast can he or she accomplish the tasks?
- Subjective satisfaction: How much does the user like using the system?

Nowadays, new concepts are arising around the concept of quality extending the original concept of usability to universal usability [Sch00], to broaden the original meaning.

The main of these concepts is accessibility which can be defined as the design of software products that are usable for the widest range of people, working in the widest range of situations.

This implies that the final user interface has to accomplish these features:

- The design should be usable and with a reasonably cost for people with different skills and tastes.
- The system should be easy to understand, independently of the user experience, knowledge, language abilities and so on.
- The system should offer the user the needed information in an effective manner, independently of the environmental conditions and user sensory abilities.
- The system should deal potential error effectively.
- The system should comfortable and efficient to use with the minimum of effort.

Achieving systems with these accessibility features is not only focus to a concrete collective of users with special disabilities, but also it is intended for people without any disabilities in some special situations. For instance, in special situations such as noisy environment people behave as a deaf; in low light conditions or in a situation which keeps your eyes occupied in other activity (i.e. driving) people behave as a blind. Thus, the introduction of these accessibility features in the final product allows the system to be used by more people in more situations.

3 Usability and Accesibility in Software Development Process

The main aim of software construction is to help users to achieve their goals in an easy way. That is, make the system to be easy to learn, easy to manage and useful, i.e. it should contain all the functionality the users need to accomplish their tasks. All in all, the basic goal is to get a computer application with a high grade of "usability" [Gou85]. As a matter of fact, usability has become a critical quality factor in the development of software systems. Currently, it is getting more and more importance.

Computer systems built with usability criteria have several benefits such as productivity improvement, cost and learning time reduction and a notable increment of final user autonomy.

Landauer [Lan95] stated that 80% of maintenance costs – what represents the 80% of software development total costs – are due to user-system interaction problems and not to technical problems. Besides, he indicates that 64% of these costs are related with usability problems.

This situation highlights the importance of usability and justifies the need to incorporate usability criteria before, during and after software systems development. This means that computer application development is a multidisciplinary activity that must incorporate human factors and ergonomic techniques with the aim of improving the work conditions of final users. This implies the participation of different kind of experts, not only computer engineers but also experts on disciplines such as psychology, ergonomics and human factors and so on.

Designing usable user interfaces is not an easy task [Mye93]. To face this problem we have to consider a great number of factors. Furthermore, we also have to deal with the design of user interfaces that will be used by users with different skills, goals or preferences. This fact is especially important for web-based user interfaces, where the heterogeneity of users is greater. Anyway, the development of traditional desktop

applications and web-based applications should be guided by a user-centered methodology to increase the success of the final application.

A user-centered methodology is characterized by the execution of a set of activities that lead to the understanding of the context of use, the users and the tasks the system will perform. Observation, prototyping, categorization and tests methods are usually used to carry out a user-centered development process [Nie93].

Usability is considered one of the software quality attributes from the first classifications of quality used in Software Engineering in the late 70's [Boe78], [McC77]. Nevertheless, historically this quality factor has been relegated with respect to other factors such as efficiency, reliability etc. when evaluating the quality of a software product. But fortunately, this point of view is changing nowadays. Perhaps the main reason for that relegation is that evaluating usability implies interaction with final users, which does not occur with the evaluation of other quality factors. Taking into account final users to measure a quality factor (usability in this case) adds new problems difficult to face.

But as stated before, this point of view is currently changing and nowadays the importance of developing software with usability criteria is unquestionable.

Some proposals have arisen trying to contribute new ideas in this field. Deborah Mayhew [May99] proposes a Usability Engineering Lifecycle in which she defines a lifecycle structured in three stages: Requirements Analysis, Design/Testing/Development and Installation. This development process follows the waterfall lifecycle. It cannot be considered as an iterative approach as it starts with Requirements analysis, then goes to the second stage of Design/Testing/Development and ends with the installation but it does not return to requirements analysis unless there is lack of functionality to be developed. This methodology is focused on user interface development. It is supposed to be complemented with the OOSE Methodology [Jac93] to cover the rest of the system development.

Constantine and Lockwood [Con99] propose a Usage-centered Design which can be considered as a method that describes the techniques to use more than a development process as defined in Software Engineering. This usage-centered design consists on a set of usability-oriented coordinated activities. They include Task modeling, Interface content modeling, etc. The most important feature they introduce is the concept of essential use cases. This is a variation of the original definition of use cases and it is the foundation of Usage-Centered Design.

Costabile [Cos01] faces the usability problem trying to integrate user-centered methods into the software process. She focuses on three basic concepts:

- Users and Task Analysis
- Iterative system design and implementation by using prototypes of increasing complexities
- Evaluation of prototypes with users.

The main idea of her proposal starts with the waterfall lifecycle. She proposes a modification of this lifecycle including two new usability oriented activities. The first one is related to users and task analysis and scenarios and user interface specifications; the second one is related to prototyping and tests. She defines the possibility of returning to a previous stage form any stage within the lifecycle. These

backwards returns within the waterfall together with the prototyping activity and tests which occur twice in the lifecycle are considered the iterative nature of her approach, although this statement is in many ways questionable.

With the aim to achieve universal access, Stephanidis [Ste01b] proposes the Unified User Interface Development Methodology. This approach seeks to convey a new perspective on the development of user interfaces, and provide a principled and systematic approach towards coping with diversity in the target user groups, tasks, and environments of user, through the user of automatic user interface adaptation. It aims to identify and enumerate possible design alternatives suitable for different users and contexts of use; identify abstractions and fuse alternatives into abstract design patterns; and rationalize the design space.

Unfortunately, a standard definition of how usability has to be included in the software lifecycle does not exist. The integration of accessibility criteria within the software process is even more unusual. Usually, accessibility is validated at the end of the software development process using the existing validation tools such as TAW, Bobby, A-PROMPT, and so on which are based on the evaluation of the accessibility guidelines defined by the WAI of the W3C Consortium. But it is difficult to find proposals trying to integrate accessibility criteria within the software development process. This paper tries to give a step forward in this sense.

4 IDEAS: A Usability Oriented Methodological Framework

As stated in the previous section, we can conclude that there are still important lacks in current software development approaches regarding the integration of usability and accessibility within the lifecycle. Some of these lacks can be summarized in lack of real iterativeness, lack of user interface modelling languages, lack of formalization and definition of the different artefacts to produce, and above all, lack of real user-centred approaches.

B. Shneiderman [Shn98] explains that, within the "usability engineering", one of the bases to reach a high quality level is using iterative design methods. These methods allow the developers to use prototypes, getting feedback from user reactions.

J. Nielsen [Nie93] states that the best way to successfully apply usability criteria is putting the maximum effort before the design process of the interface layout starts. With this concern, the most important thing is to make a previous study that allows the designer to "know the users". This knowledge is based not only on the study of users' characteristics or individual abilities, but also it is necessary to know the tasks they need to carry out. In this sense, it is very important that users could perform evaluations (usability tests) upon the prototype. With this practice we can obtain improvements in next versions of the product.

For the reasons stated before, our proposal includes high-level models which allow us to "know the users". For this matter, IDEAS includes use case, task and user models. The first ones allow us to describe user tasks accurately. The last one describes the individual characteristics of the different kind of users interacting with the system.

After knowing the users and the tasks they perform, it is time to set the goals or basic usability and accessibility criteria the system must reach. The definition of these criteria will allow us to perform the usability tests associated to such criteria further on. In this stage, a quality model gathering these usability criteria and metrics is defined.

In last phases of the development when designing the presentation model, the designer, considering the quality model previously defined, must take into account different heuristics and usability guidelines and accessibility criteria to improve the quality of the layouts. Starting from the presentation models, the user interface prototype is generated. Final users evaluate these prototypes with the usability and accessibility tests based on the quality model previously defined.

Finally, all the previously stated phases and models are incorporated into an iterative design process. This way of doing allows performing a feedback process by introducing the information collected in the tests performed by the users in the user interface development. This way, the usability of the designed interfaces improves notably.

In figure 3, this iterative process integrating usability is depicted.

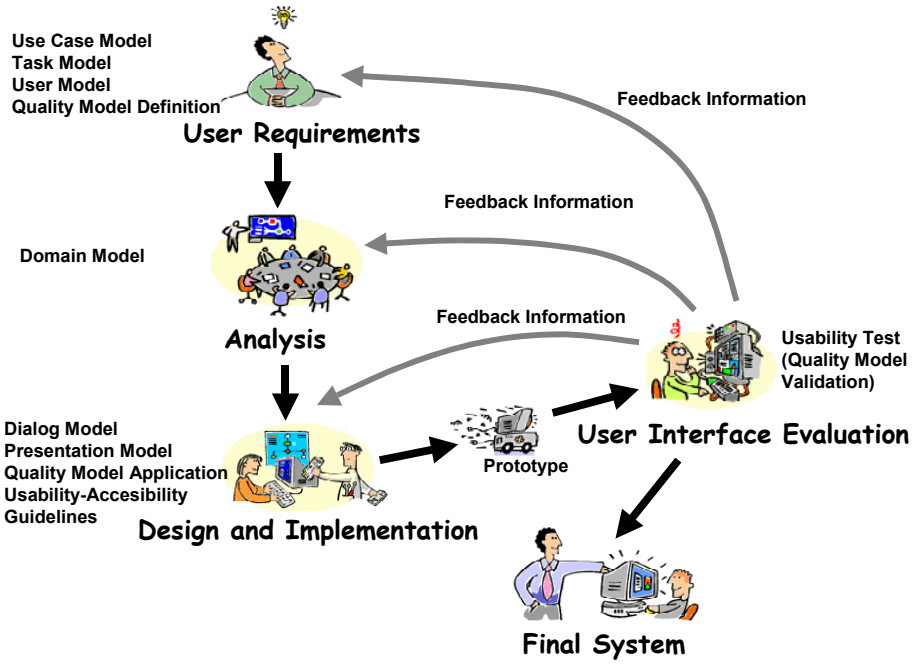


Fig. 3. Iterative development process

The aim of this software process is the integration of a user interface model including usability and accessibility criteria within an object oriented software production environment. The user interface specification process is tackled in parallel to the application development, and according to the common principles of Model-

based User Interface Development Environments [Sch96]. This development process is depicted in more detail in figure 4.

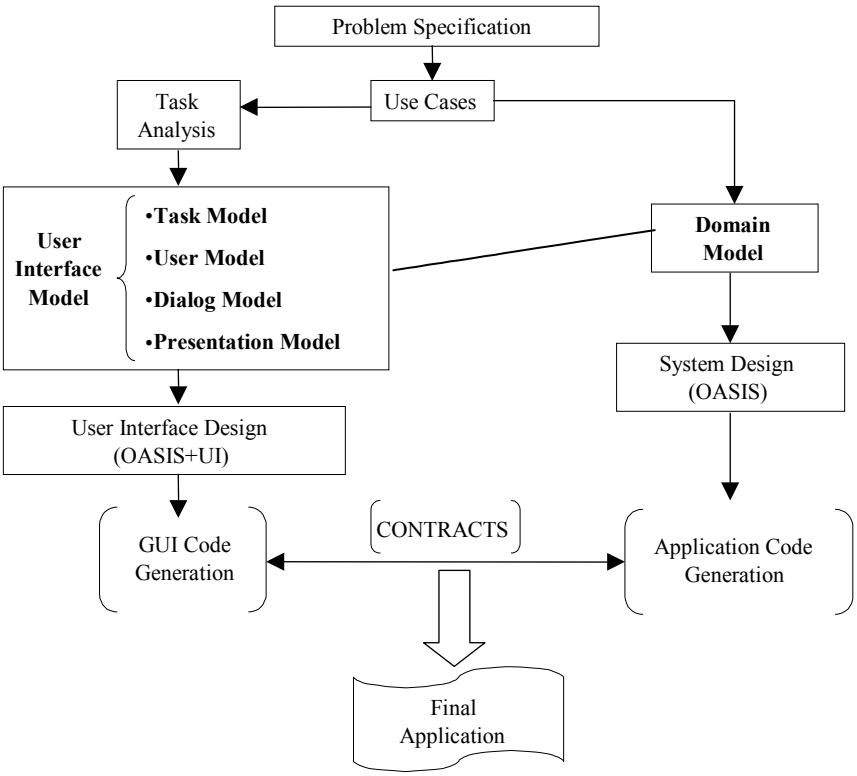


Fig. 4. IDEAS: Interface Development Environment within OASIS

IDEAS aims to be a semi-automatic user interface development system integrated within the framework of the object oriented model OASIS [Let98] to support the automatic production of high-quality user interfaces including usability criteria. This is possible due to the fact that this environment is based on declarative models, i.e., all aspects of the user interface design are represented using declarative models. This approach offers several benefits:

- User-centered development life cycle.
- Centralized user interface specification.
- Design tools for an automated and interactive development.
- Reuse of the user interface designs.

It is important to highlight that this user interface development process is independent of the final platform where the Graphical User Interface (GUI) is going to run. It could be a web interface, a personal computer interface, a PDA or any other

platform, as a decision of this kind is only made in the last phase of the methodology, i.e., the same design may be used for different implementations in different devices.

As shown in figure 4, the development process is split into two parts: On the one hand the development of the GUI is performed and on the other hand the application itself is developed.

These two independent and parallel developments take as starting point the same information: a set of use cases providing the requirements of the domain activity to be supported. Starting from this common information, on the one hand a domain model is created, that is, a conceptual object model that describes the objects identified within the system together with the relations among them.

But on the other hand, and in order to develop the GUI, we need additional information apart from the one gathered by the use cases. Use cases are not enough to describe user tasks from the point of view of the final user. Use cases describe the systems in terms of functional requirements; nevertheless, one of their main deficiencies is that they do not capture non-functional requirements, such as reliability, efficiency, maintainability and above all, usability, which undoubtedly are critical factors for final users to accept a computer application. Use cases do not capture user needs either, which are essential to develop user interfaces. For these reasons, task analysis techniques are included at this point. Starting from task analysis, the User Interface Model is built in the way described below. The relationship between use cases and the tasks identified with the task analysis technique is one to many, this is, one use case may correspond to one or many tasks. So, the system functionality described by a use case is equivalent to one or many user tasks.

The approach we propose is to take a use-case-driven development process adding task analysis techniques to enable the specification of the user interface in a usable way. These techniques mainly help to define who the users of the system are, what tasks should they perform, what the main goal of those tasks is, and so on.

On the other hand (the right side in figure 4), the design of the system supporting the solution is developed. In many ways these two designs (the user interface and the application) specify two separate but interdependent systems.

In IDEAS, the declarative models composing the user interface model are the following: Task, User, Domain, Dialogue and Presentation Models. In next section, these models will be defined following the different stages of the development process.

5 Development Process

The user interface development process proposed in IDEAS is depicted in figure 5.

At Requirements Level three models are created: the Use Case Model, the Task Model and the User Model.

Firstly, the use cases technique is used. In this first stage and taking into account the use cases, the different kinds of users are identified. Subsequently, each use case is refined and the business rules, entities and actors that participate in it are specified.

The Task Model defines the ordered set of activities and actions the user has to perform to achieve a concrete purpose or goal. Artifacts are essential for task development. They are modeled as objects and represented in the domain model. There exist a strong relation between the task model and the domain model.

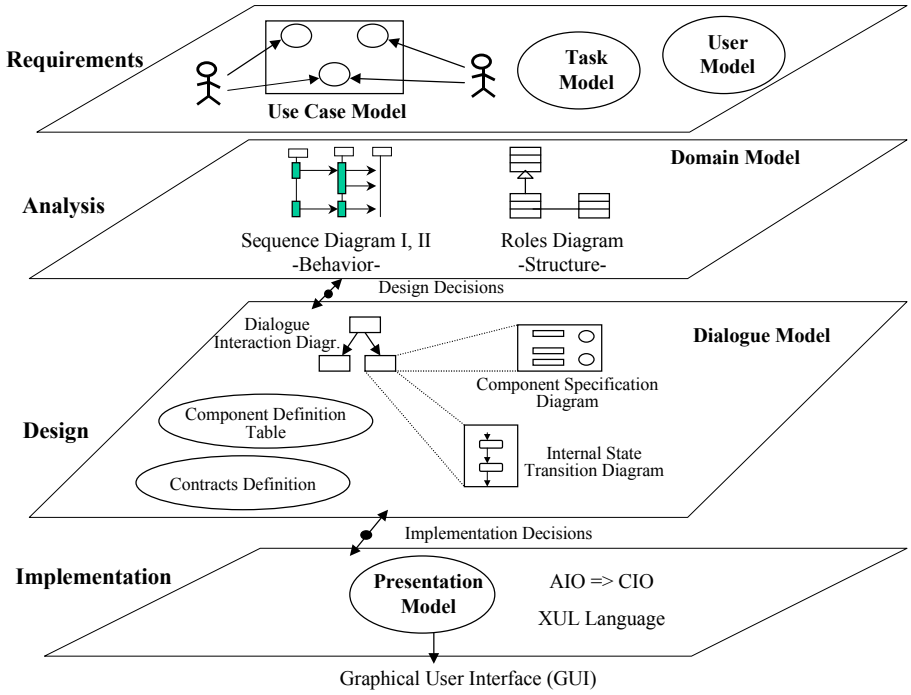


Fig. 5. User Interface Development Process

According to [Sto95], a task description should include a goal, a non-empty set of actions necessary to achieve the goal, a plan for the selection of actions and a model of the artifact involved in the task development. This description of what a task is and must contain can be used to construct a task definition language. We propose a template, based on the one proposed by [Coc97], to describe in natural language all these issues.

The User Model describes the characteristics of the different types of users. The purpose of this model is to support the creation of individual and personalized user interfaces. Firstly, it defines, for each kind of user, the set of tasks he/she can perform. Secondly, for each kind of user in relation with a concrete task, a projection on the actions within the task that he/she can perform is established. And finally, depending on the user's particular characteristics (child, adult, handicapped...), the information and the interaction established by the Dialog Model to show the information contained in the Domain Model is adapted to the user. All these characteristics are used to define the properties to be included in the quality model.

At analysis level the Domain Model is performed. This model consists of two diagrams. The first one is the Sequence Diagram a slight variation based on the

original UML sequence diagrams, which defines the system behavior. The second one is the Roles Model, which defines the structure of the classes that take part in the associated sequence diagram together with the relationships among these classes, specifying the role of each one of them.

At design level, the Dialogue Model is performed [Loz00]. All the models that have been generated up to now do not contain any graphical aspect of the final user interface. It is from now on that these issues start to be addressed and the way in which the user-system interaction will be performed is especially important. The dialogue model includes the generation of four different kinds of diagrams. Herein and due to space constraints, we highlight the Dialogue Structure Diagram and the Component Specification Diagram. The first diagram specifies the user interface behavior. It represents the windows and dialogues the user needs to complete all the tasks he/she requires from the system, and the user selections to pass from one window to another. For the generation of this diagram, we have to take into account the sequence diagram obtained in the analysis phase. The second diagram entails establishing, for every one of the windows and dialogues obtained in the first diagram, the set of control and visualization tools needed to give the user the functionality he requires performing the corresponding task. This diagram defines the state of the user interface. At this level the user interface is designed by means of Abstract Interaction Objects (AIOS), so that we do not have to decide yet the user interface concrete widgets. This decision is postponed to the next step considering the final implementation platform.

At implementation level the Presentation Model is performed. This model describes the concrete interaction objects (CIOs) composing the final GUI, its design characteristics and visual dependencies among them. These CIOs are defined taking into account implementation decisions, the final implementation platform and according to the style guide followed. The final GUI has a static and a dynamic part. The first one is the standard widgets presentation and the second one shows the data dependent on the application that change at run time. In IDEAS, the final GUI generation is performed by using XUL , an XML based language, in order to make it as much independent as possible from the final platform where the application is going to run. The use of different abstract levels (AIOs-CIOs) allows us to attain the "technology variety" challenge stated by Shneiderman [Shn00] as this way of doing allows us to perform different implementation under different platforms starting from the same design.

Next we show a fragment of a case study developed with the tool IDEAS-CASE [Loz02] that supports the process explained before. The example is based on a web application for a Conference Review System. Figure 6 shows the use case diagram for the Conference Review System, showing also the different development stages in which the tool is structured.

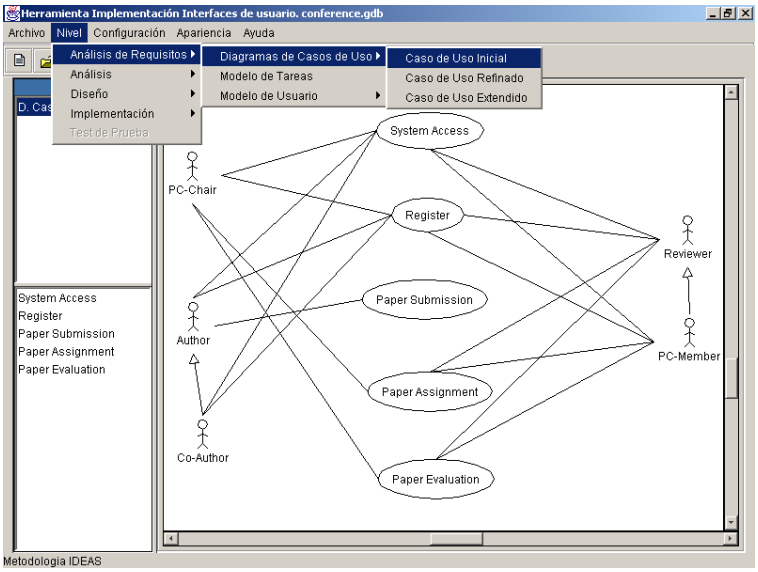


Fig. 6. IDEAS Tool: Initial Use Case Diagram

Following the development process proposed within IDEAS, at design level we finally get the Component Specification Diagram showed in figure 7. This diagram shows the Abstract Interaction Objects (AIO) the user needs to perform the task of logging into the system.

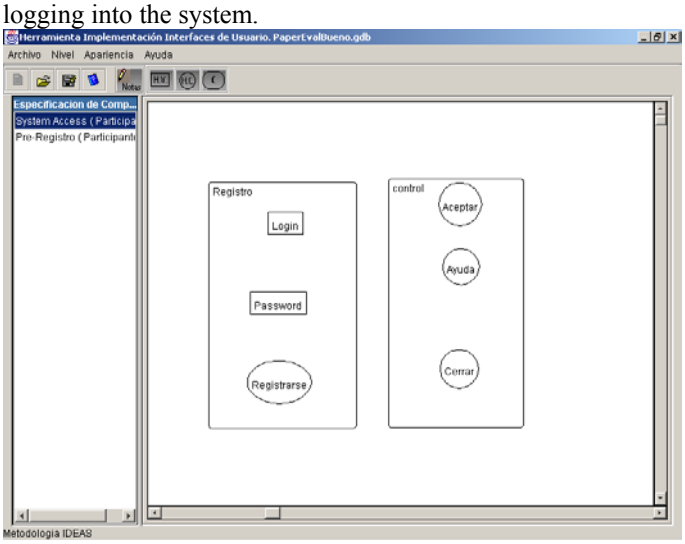


Fig. 7. Component Specification Diagram (AIOs) corresponding to the "log into the system" task

Starting from this diagram, the AIO are translated into CIO as depicted in figure 8, automatically generating the final GUI shown in figure 9.

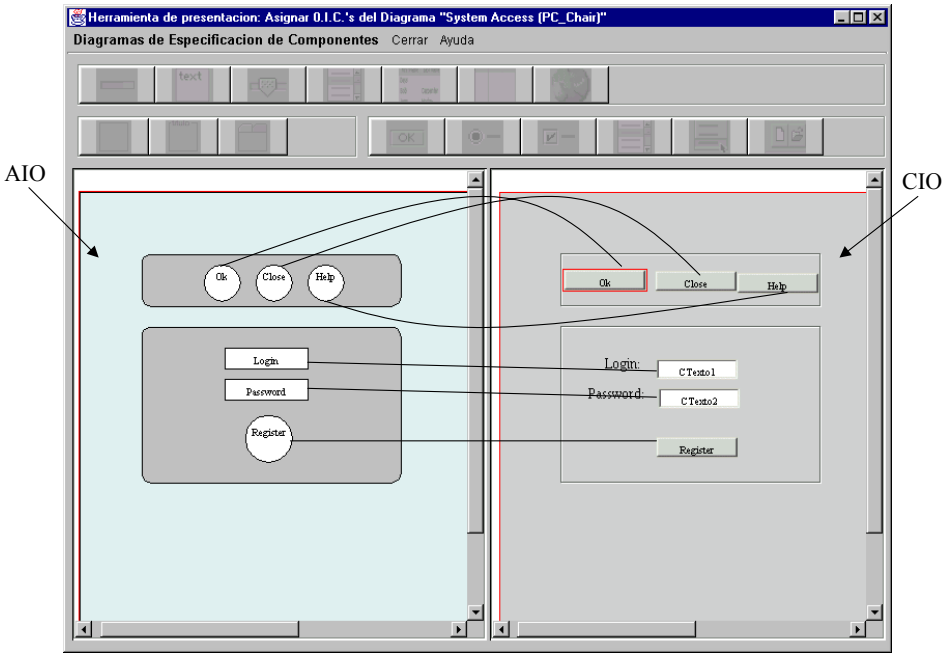


Fig. 8. Conversion from AIO to CIO

Fig. 9. Final Graphical User Interface

6 Conclusions and Future Works

In this paper we have tackled the problem of improving the user interface quality to get closer to the concept of Universal Access and allow so that the highest number of citizens in the highest number of conditions can access the new Information Technologies.

To achieve this general goal it is essential to incorporate usability and accessibility criteria within the software development process. But, currently there are very few proposals including these criteria accurately. We detect the need to advance in this research topic. In this sense, we have tried to give a step forward and we propose a methodological framework incorporating a quality model in its process model as a mechanism to define and validate usability and accessibility criteria. This development process not only allows us to generate adaptable user interfaces depending on the particular characteristics of the different kinds of user considered in the proposed User Model, but also the definition of different abstraction levels allows us to generate different implementations from the same abstract specification.

But there are still lots of aspects left to improve. In this current proposal we consider a pre-defined and pre-known set of users defined at requirements level. With this information we can obtain adaptable user interfaces at design time, but actually the potential range of users is unknown. We should consider this fact and take into account these potential users if we want really to attain a Universal Access. In this sense, it is essential to achieve adaptative user interfaces at run-time incorporating new agent-based mechanisms to add adaptivity and personalize the systems depending on the different users. Currently we are working on these aspects to improve the usability oriented methodological framework.

Acknowledgement

This work is supported in part by the Spanish Junta de Comunidades de Castilla-La Mancha PBC-03-003 grant.

References

[Bas94] Basili, V., Caldiera, G., and Dieter H., (1994). The Goal Question Metric Approach, in Encyclopedia of Software Engineering, Two Volume Set, Gianluigi Caldiera and Dieter H. Rombach, Eds. New York City: John Wiley and Sons, Inc., pp. 528532, 1994.

[Boe78] Boehm, B. Characteristics of Software Quality. North Holland Publishing Co., New York, 1978.

[Bra02] Brajnik, G. Quality Models based on Automatic Webtesting. Presented at CHI2002 workshop Automatically evaluating usability of Web Sites, Minneapolis (MN), April 2002, www.acm.org/sigchi/chi2002. 2002.

[Coc97] A. Cockburn: Structuring Use Cases with Goals. Journal of Object-Oriented Programming, 10(5) y 10(7), 1997. http://members.aol.com/acockburn/papers/usecases.htm

[Con99] Constantine, L. and Lockwood, L. Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design. Addison-Wesley, New York, 1999.

[Cos01] Costabile, M.F. Usability in the Software Life Cycle. Handbook of Software Engineering and Knowledge Engineering. World Scientific Publishing, pp. 179-192. Singapore, 2001.
[Gou85] J. Gould, C. Lewis: Designing for Usability – Key Principles and What Designers Think. Commun. ACM, Vol 28, 1985, pp.300-311.
[IEEE-1061] IEEE Std. 1061-1998 IEEE Standard for a Software Quality Metrics Methodology. 1998.
[ISO00] ISO/IEC 9126-1. Software Engineering. Product quality. Part 1: Quality model. 2000
[ISO01] ISO/IEC 9126-4. Software Engineering. Product quality. Part 1: Quality in use metrics. 2001
[ISO98] ISO/DIS 9241: Ergonomic Requirements for Office Work with Visual Display Terminals (VDT's). Geneva: International Standards Organization. 1998.
[Lan95] Landauer, T.K. "The Trouble with Computers: Usefulness, Usability and Productivity. MIT Press, 1995.
[Let98] P. Letelier, I. Ramos, P. Sanchez, O. Pastor. OASIS versión 3.0: A Formal Approach for Object Oriented Conceptual Modelling. SPUPV-98.4011. Edited by the Technical University of Valencia, Spain, 1998.
[Loz00] M. Lozano, I. Ramos, P. González. User Interface Specification and Modeling in an Object Oriented Environment for Automatic Software Development. 34th International Conference on Technology of Object-Oriented Languages and Systems. TOOLS-USA 2000. Santa Barbara, CA. 30 July - 3 August, 2000.
[Loz02] M. Lozano, J.P. Molina, F. Montero, P. González, I. Ramos. A Graphical User Interface Development Tool. Proceedings of the 6th Anual Conference on Human Computer Interaction (HCI'02) Ref.: ISBN: 1-902505-48-4, Londres, Inglaterra, 2002.
[May99] Mayhew, D.J. The Usability Engineering Lifecycle. Morgan Kaufmann, San Francisco, CA. 1999
[McC77] McCall, J.A., Richards, P.K., Walters, G.F. Factors in Software Quality". Vol. 1, 2, 3, AD/A-079-014/015/055. National Tech. Information Service, Springfield, 1997.
[Mye93] B.A. Myers: Why are Human-Computer Interfaces Difficult to Design and Implement. Tech. Report CMU-CS-93-183, CMU, Jul 1993.
[Nie93] J. Nielsen. Usability Engineering. Morgan Kaufmann, 1993.
[Ols02] Olsina, L., Rossi, G. Measuring Web Application Quality with WebQEM. IEEE Multimedia. October-December. pp. 20 – 29. 2002.
[Sch96] Schlungbaum, E.. Model-Based User Interface Software Tools - Current State of Declarative Models. Technical Report 96-30, Graphics, Visualization and Usability Center, Georgia Institute of Technology, 1996.
[Shn98] B. Shneiderman. Designing the User Interface. Strategies for Effective Human-Computer Interaction. Addison Wesley, 1998.
[Sch00] B. Shneiderman. Universal Access: Pushing human-computer interaction research to empower every citizen. CS-TR-4043. CommunACM 43(5):84-91.
[Ste01a] Stephanidis, C. User Interfaces for all: new perspectives into HCI. In User Interfaces for all- concepts, methods and tools. Lawrence Erlbaum Associates, Mahwah, NJ. pp. 3-17, 2001.
[Ste01b] Stephanidis, C., Savidis, A. Universal Access in the Information Society: Methods, Tools and Interaction Technologies. UAIS 1: 40-55. 2001.
[Sto95] G. Storrs: The Notion of Task in Human-Computer Interaction. In: People and Computers X. Proceeding British HCI'95 (Huddersfield UK, Agosto 1995). Cambridge: Cambridge University Press, 1995, 357-365.

Soporte Formal para Entornos Visuales de Modelado ♦

Artur Boronat, José Á. Carsí, Isidro Ramos, Julián Pedrós

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
C/Camino de Vera s/n
E-46022 Valencia- España
{aboronat | pcarsi | iramos | jpedros}@dsic.upv.es

Abstract. Las técnicas formales de modelado aportan propiedades como la precisión y la corrección, y permiten dar soporte automatizado al proceso de desarrollo de software. MOMENT es una herramienta de gestión de modelos que permite definirlos siguiendo una aproximación algebraica, y que se beneficia de las propiedades de precisión que aporta esta última. Sin embargo, la complejidad de estas técnicas suele provocar rechazo a la hora de utilizarlas en la práctica. Una alternativa para sortear este inconveniente consiste en la asociación de metáforas gráficas cercanas al usuario que oculten la complejidad asociada a las especificaciones algebraicas de forma que los usuarios trabajen en un ambiente sencillo y agradable. En este artículo se presenta una arquitectura que integra MOMENT en herramientas CASE que posean un entorno gráfico, pudiendo así utilizar sus capacidades visuales en la definición de los modelos soportados por MOMENT. Se presenta la implementación de esta arquitectura para una herramienta CASE concreta, y cómo se define la vista gráfica de un metamodelo definido algebraicamente utilizando su interfaz.

Palabras clave: gestión de modelos, herramientas CASE, modelado visual, extensibilidad, especificaciones algebraicas.

1. Introducción

Los lenguajes formales de modelado normalmente son difíciles de usar, incluso para aquellos que les podrían sacar un buen rendimiento, como ingenieros de software y desarrolladores de aplicaciones. Sin embargo, a medida que los problemas a resolver crecen en magnitud y en complejidad, las técnicas de especificación formal toman relevancia para garantizar el desarrollo de un producto software de calidad y correcto. Una aproximación para reducir la dificultad de uso de lenguajes de especificación formal consiste en aplicar una notación gráfica que permita representar de forma sencilla los conceptos de las ontologías que el usuario debe utilizar.

MOMENT [1] es una herramienta que permite definir modelos como especificaciones algebraicas de una teoría expresada en lógica ecuacional condicional

♦ Este artículo ha sido financiado por el Proyecto Nacional de Investigación, Desarrollo e Innovación DYNAMICA TIC 2003-07804-C05-01 y el Proyecto Nacional PBC-03-00 de "Metodologías de desarrollo de interfaces de usuario dinámicas".

[22]. MOMENT proporciona un mecanismo para transformar modelos basados en el álgebra que utiliza la herramienta, basándose en el soporte formal y automático que proporcionan los sistemas de reescritura de términos [3], como Maude, CafeOBJ, etc. Este mecanismo de transformación permite automatizar gran cantidad de procesos software, como por ejemplo, los estudiados en el contexto *Model-Driven Architecture* (MDA) [23], que permiten entre otras cosas generar modelos específicos de plataforma partiendo de modelos independientes de plataforma.

Con el fin de aprovechar las propiedades de MOMENT, se presenta una arquitectura para integrar su funcionalidad en un entorno de modelado visual. De esta forma, se permite asociar especificaciones algebraicas a notaciones visuales, de manera que cuando el usuario utilice dichos gráficos para construir un modelo, MOMENT define automáticamente la especificación algebraica asociada el modelo diseñado, de forma transparente al usuario. Un aspecto importante de esta aproximación es la posibilidad de especificar algebraicamente cualquier notación gráfica. De esta forma, las propiedades de la herramienta se pueden aplicar a diferentes aproximaciones de modelado: independientes de dominio, como UML, o bien específicas a dominio. Esta arquitectura ha sido implementada para la herramienta MS Visio 2003 [4], porque proporciona buenas propiedades de extensibilidad, tanto en las facilidades gráficas, ya que dispone de un editor gráfico, como en la funcionalidad interna. De esta manera, el usuario puede especificar modelos de forma algebraica basándose en una metáfora visual conocida a través de una interfaz sencilla.

En esta sección se han introducido las ventajas de asociar una representación gráfica a un determinado formalismo. En la sección 2 se indica la estructura de la plataforma de gestión de modelos MOMENT y los elementos principales que se utilizan en ella. En la sección 3, se describe los elementos principales del entorno gráfico de una herramienta visual de modelado. En la sección 4, se indica la arquitectura que hemos utilizado para añadir el soporte formal a una herramienta CASE, identificando las asociaciones entre los elementos de su editor gráfico y los sorts (o nombres de tipos) de la teoría algebraica utilizada en MOMENT, y se describe el mecanismo que permite asociar una metáfora gráfica a un metamodelo especificado algebraicamente en MOMENT. En la sección 5 se indica cómo se realiza la especificación algebraica de un modelo de forma automática. En la sección 6 se discuten trabajos relacionados, y finalmente, en la sección 7, se resumen una serie de conclusiones y trabajos futuros.

2. La plataforma MOMENT (MOdel manageMENT)

MOMENT es una plataforma de gestión de modelos implementada algebraicamente usando el modelo computacional l-cálculo mediante el lenguaje funcional F# [5]. MOMENT utiliza también el entorno CafeObj [6] como sistema de reescritura de términos para realizar transformaciones automáticas de modelos entre diferentes metamodelos (transformaciones intermodelo) o de un mismo metamodelo (transformaciones intramodelo).

La plataforma posee cuatro niveles de abstracción, siguiendo la cultura de metamodelado presentada en el estándar MOF [7], aunque no implementa directamente su lenguaje abstracto para definir metamodelos. Cada nivel está formado por un conjunto de esquemas que a su vez están constituidos por un conjunto de conceptos y propiedades. Los conceptos permiten representar entidades descriptibles en un dominio determinado, y las propiedades establecen dichas descripciones, pudiendo asociar dos conceptos mediante una propiedad o bien describir un concepto mediante una propiedad que contiene un valor básico.

Los niveles que consituyen la plataforma MOMENT son los siguientes:

- Nivel M3: contiene los elementos (conceptos y propiedades) necesarios para poder describir cualquier metamodelo en el siguiente nivel. Es el nivel más abstracto de la plataforma.
- Nivel M2: sus esquemas constituyen metamodelos, donde sus respectivos conceptos y propiedades indican como definir un modelo en el siguiente nivel.
- Nivel M1: sus esquemas representan modelos de un determinado metamodelo del nivel M2. Los conceptos y propiedades que forman un modelo permiten definir información en el siguiente nivel.
- Nivel M0: sus esquemas representan instanciaciones de modelos del nivel M1 que contienen datos concretos.

3. Elementos principales del entorno visual de herramientas CASE de modelado

El área de trabajo de las herramientas CASE de modelado visual tienen cuatro elementos en común, que son descritos a continuación:

- *Figuras*: Símbolos que pueden ser prediseñados o pueden ser de creación propia, ajustándose a las necesidades del modelo que el usuario necesita representar. Dichos símbolos pueden contener información de valor añadido mediante propiedades.
- *Hoja de dibujo*: Zona de trabajo donde se depositan (pegan) las figuras, constituyendo un diagrama o modelo gráfico concreto.
- *Primitivas gráficas*: Distintos tipos de figuras que se pueden definir en la hoja de dibujo de la herramienta CASE. Una primitiva gráfica permite definir una figura en la hoja de dibujo. Poseen unos atributos o propiedades por defecto que definen las características de sus figuras.
- *Vistas gráficas*: Plantillas que agrupan a las primitivas gráficas según el modelo o tipo de diagrama al que pertenecen, encapsulando la ontología gráfica que permite definir un modelo.

MS Visio 2003 es una herramienta visual de modelado fácilmente personalizable y extensible mediante la tecnología .NET. En el vocabulario propio de la herramienta, los elementos que hemos identificado anteriormente de forma genérica reciben los siguientes nombres: las figuras son denominadas *shapes*; la hoja de dibujo *shapesheet*; una primitiva gráfica recibe el nombre de *master*; y la vista que agrupa una serie de primitivas gráficas recibe el nombre de *stencil*.

La personalización del entorno visual de Visio se realiza mediante add-ons, es decir, conjuntos de *stencils* que proporcionan la información que constituye un metamodelo completo. La extensión de la herramienta se consigue mediante add-ins, que son módulos que permiten añadir funcionalidad a la herramienta. Dada la facilidad de extensión aportada por Visio 2003, ha sido la herramienta escogida para construir la interfaz gráfica de usuario de MOMENT.

4. Arquitectura de la solución

Partiendo de la plataforma formal de gestión de modelos MOMENT, se ha desarrollado un módulo, llamado *MOMENT Integrator*, que permite asociar una metáfora gráfica a un metamodelo formal mediante la interfaz de herramientas CASE con entornos visuales que proporcionen cierto grado de extensibilidad, como MS Visio 2003.

4.1. Arquitectura

En la Fig. 1, se muestra la arquitectura que permite integrar la plataforma MOMENT en una herramienta CASE con un entorno gráfico de forma genérica. Aunque se ha contextualizado su implementación en el entorno Visio 2003, se está pensando en desarrollar la misma arquitectura para integrar MOMENT en la herramienta de desarrollo Eclipse aprovechando su entorno gráfico Graphical Editor Framework [20]. La arquitectura utilizada es la tradicional separación en tres capas: una interfaz que permite representar gráficamente metamodelos y modelos, la funcionalidad que permite asociar la metáfora gráfica a un metamodelo especificado algebraicamente y la capa de persistencia en la que se almacena toda la información utilizada.

En la interfaz gráfica, la herramienta CASE ofrece la funcionalidad necesaria para definir los gráficos que se van a utilizar en un metamodelo, y permite acceder a la funcionalidad del módulo *MOMENT Integrator*.

En la capa de funcionalidad, el módulo *MOMENT Integrator* permite definir asociaciones entre los elementos gráficos que se han definido mediante el editor gráfico de la herramienta CASE y los términos algebraicos mediante los cuales MOMENT representa un metamodelo. Como veremos en la siguiente sección, dichas asociaciones se almacenan en la propia plataforma como instancias de clases UML a través de la librería MomentUMLSupport.

En la capa de persistencia se pueden distinguir dos unidades principales de almacenamiento: la de los elementos gráficos de la herramienta CASE y la de la plataforma MOMENT.

Las herramientas CASE suelen almacenar la información gráfica mediante una serie de ficheros. En el caso específico de la herramienta Visio, los modelos gráficos se almacenan mediante dos tipos de ficheros. Los ficheros con extensión *.vss* almacenan los documentos que se crean, o sea el área dibujada con figuras (*shapesheet*), junto con las plantillas o templates a partir de las cuales se ha creado el diagrama o modelo. Los ficheros con extensión *.vst* almacenan las plantillas con *masters* (*stencils*). Por defecto, Visio ya incorpora un gran número de plantillas con

todo tipo de formas para todo tipo de modelos, pero un usuario puede crearse y guardar sus propias plantillas con una colección de formas de otras plantillas o con formas creadas por el mismo.

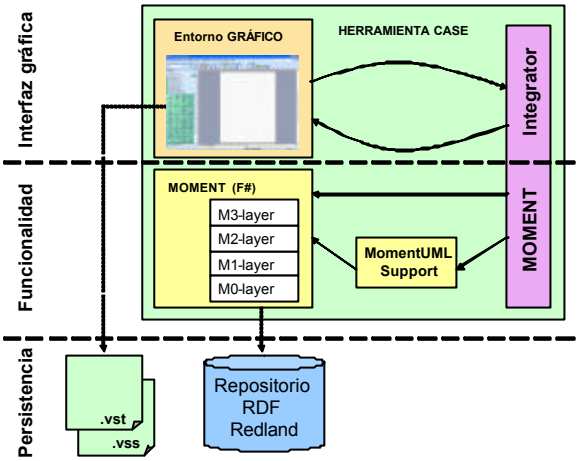


Fig. 1. Arquitectura de la integración de MOMENT en una herramienta CASE.

Por otro lado, MOMENT almacena toda la información en un repositorio RDF puesto que los conceptos y propiedades manejados en la plataforma son un reflejo de los recursos y propiedades RDF. El repositorio utilizado es Redland [19] y permite almacenar la información sobre esquemas de todos los niveles de la plataforma y asociaciones entre elementos gráficos del entorno visual y términos de MOMENT, explotando las facilidades que nos ofrece el entorno gráfico de la herramienta CASE.

4.2. Diseño de la capa de funcionalidad

Mediante el módulo *MOMENT Integrator*, los elementos gráficos de la herramienta CASE se vinculan con los elementos de la plataforma MOMENT. Cómo nuestro objetivo consiste en la definición de la metáfora gráfica asociada a un metamodelo, nos centramos en el uso de los esquemas del nivel M2 de la plataforma, que son los metamodelos. De manera que, un esquema de este nivel se asocia con una vista gráfica (*stencil* en Visio). Por otro lado, para representar gráficamente los conceptos y las propiedades de los esquemas que constituyen un metamodelo en el nivel M2 de la plataforma, hacemos uso de las primitivas gráficas que componen la vista gráfica elegida (*masters* en el contexto de Visio).

Para asociar la información de cada elemento de un metamodelo específico de MOMENT a las figuras gráficas que proporciona el entorno visual, se ha definido una agregación, expresada en notación UML en la Fig. 2. En esta agregación participan las siguientes clases:

– *GraphicViewWrapper*: es la clase agregada y encapsula información sobre una vista gráfica determinada y sobre el esquema correspondiente del nivel M2 de la plataforma. Es en esta clase, por tanto, donde se asocia el esquema del metamodelo a la plantilla gráfica. Una instancia de la clase *GraphicViewWrapper* está formado

por un conjunto de instancias de la clase *GraphicPrimitiveWrapper* que definen las primitivas gráficas del metamodelo.

La información que contiene esta clase es el identificador o nombre del contenedor, el de la vista gráfica (*stencil* en el contexto de Visio) al que se le asocia el metamodelo junto con la ruta completa donde se encuentra el fichero en el que se almacena la vista gráfica, y finalmente los atributos que identifican la definición del metamodelo en la plataforma, es decir, los identificadores del nivel y del esquema. El tipo de datos *Repository* permite abstraer el mecanismo de almacenamiento de una herramienta CASE específica en el módulo coordinador *MOMENT Integrator*.

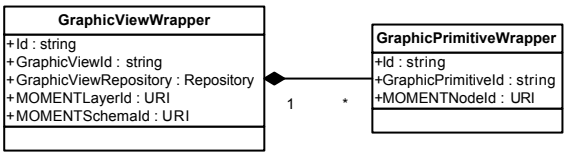


Fig. 2. Diagrama de clases UML que modela las asociaciones de elementos gráficos con elementos de la plataforma MOMENT.

– *MasterWrapperClass*: es la clase componente que encapsula la información sobre una primitiva gráfica específica (*master* en el contexto de Visio) y el concepto correspondiente de un metamodelo de la plataforma MOMENT. Este concepto puede disponer de una serie de propiedades, definidas como términos en MOMENT.

Los atributos de esta clase son un identificador o nombre para el contenedor, el identificador de la primitiva gráfica que pertenece a la vista gráfica especificada y el identificador del nodo MOMENT (concepto o propiedad) que se asocia.

La asociación entre elementos gráficos de la herramienta CASE con elementos de la plataforma definida en el diagrama de clases UML de la Fig. 2, se almacena utilizando la propia plataforma MOMENT como repositorio orientado a objetos, puesto que permite definir modelos UML y sus extensiones.

Para llegar a tal fin, se aprovecha la parte del metamodelo UML, especificado como un esquema en el nivel M2 de la plataforma, que permite la definición de clases y de asociaciones. Por tanto, se ha definido el modelo UML de la Fig. 2 como un esquema del nivel M1 de la plataforma. De esta manera, MOMENT ya dispone de la información necesaria para almacenar instancias de las clases del diagrama de clases definido.

Para facilitar el uso de MOMENT como repositorio orientado a objetos, se ha desarrollado una API que se comporta como la interfaz directa entre la plataforma de gestión de modelos y el módulo *MOMENT Integrator*. Esta librería, llamada MomentUMLSupport, ha sido desarrollada en C# y utiliza las funciones de un módulo F#, llamado UML.ml, que utiliza la funcionalidad de MOMENT para insertar instancias de las dases de un modelo del nivel M1, en una instanciación de modelo del nivel M0. La Fig. 3 muestra la estructura de la librería MomentUMLSupport.

El API de la librería MomentUMLSupport permite manipular instacias de clases de cualquier modelo UML, involucrando la instanciación, consulta o destrucción de objetos en el nivel M0 de la plataforma. De manera que cuando se instancia una clase mediante esta librería, se está almacenando la información en la plataforma.

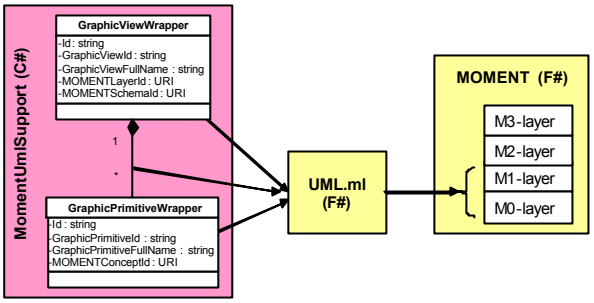


Fig. 3. Estructura de la librería MomentUMLSupport

4.3. Definición de la metáfora gráfica de un metamodelo

Para construir un metamodelo mediante una herramienta CASE asociamos un metamodelo MOMENT (un esquema del nivel M2) a una vista gráfica. Una vez definido el significado de esa vista, se especifica cada una de sus primitivas gráficas con conceptos y propiedades de un esquema del nivel M2 de la plataforma, completando la metáfora gráfica asociada al metamodelo. A continuación se detallan estos dos pasos en el contexto de la implementación que se ha realizado para la herramienta MS Visio 2003:

1. En primer lugar, para definir el *stencil* se accede a la interfaz de definición que se muestra en la Fig. 4, donde se selecciona el *stencil* que se desea asociar y el esquema del nivel M2 de la plataforma que define el metamodelo.
2. Una vez asociada una vista gráfica a un metamodelo MOMENT mediante una instancia *GraphicViewWrapper*, ya sea por haber seguido los pasos anteriores o por haber cargado la plataforma MOMENT de un repositorio Redland, se definen gráficamente los conceptos y propiedades que posee el metamodelo en MOMENT.

   Para ello se accede a la interfaz de la Fig. 5, donde se selecciona un metamodelo definido gráficamente. Acto seguido, se selecciona una primitiva gráfica de la vista seleccionada y un nodo (concepto o propiedad) del metamodelo MOMENT especificado.

   Después de haber definido una primitiva gráfica, se pueden consultar las propiedades ocultas que describen al nodo MOMENT, es decir, aquéllas que no están definidas gráficamente. La información de esas propiedades aparece en la lista inferior, y además se puede navegar recursivamente por las propiedades mediante el árbol situado en la parte izquierda inferior de la ventana.

4.4. Definición gráfica de especificaciones algebraicas de modelos

Una vez se ha definido un metamodelo en la herramienta CASE, se puede definir un modelo de la forma convencional mediante la técnica *drag-and-drop* característica de este tipo de entornos de modelado, soltando primitivas gráficas de la vista gráfica sobre la hoja de dibujo, creando una nueva figura. El módulo *MOMENT Integrator*

también enriquece esta funcionalidad definiendo los términos del esquema correspondiente del nivel M1 de forma automática y transparente al usuario.

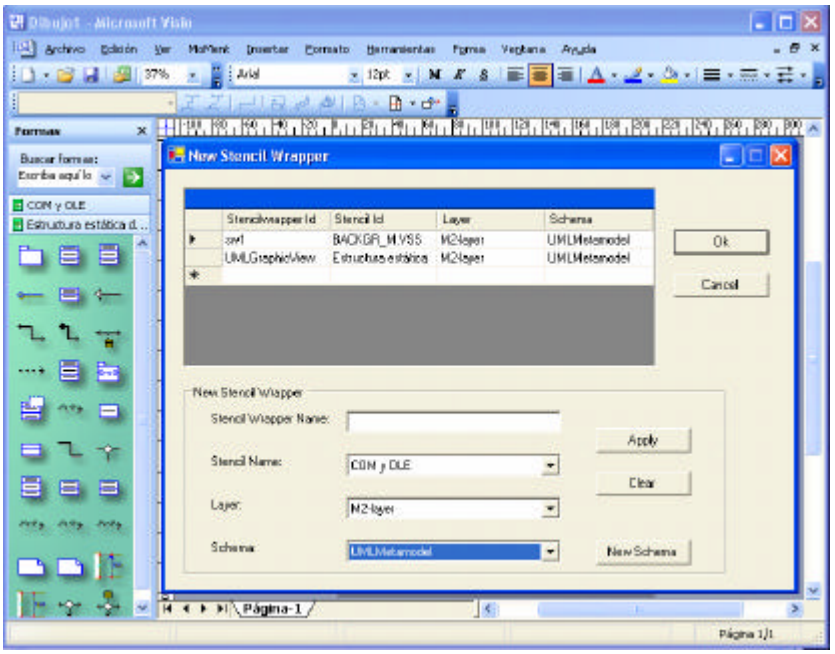


Fig. 4. Creación de un metamodelo mediante VisioMoment

En primer lugar, para instanciar un metamodelo gráfico especificado algebraicamente mediante MOMENT, se debe crear una hoja de figuras que constituye el contenedor que albergará las figuras que representan gráficamente un modelo determinado. A continuación, se pueden definir los elementos gráficos que constituyen el modelo soltando primitivas gráficas sobre la hoja de figuras. Dependiendo de la especificación algebraica subyacente a una primitiva gráfica se definen diferentes comportamientos:

- Cuando la primitiva gráfica está asociada a un concepto, la creación de una nueva figura implica la definición de un concepto en el esquema correpondiente del nivel M1 de la plataforma. Además, se instancian todas las propiedades asignadas al concepto padre, que tengan definida una restricción sobre la cardinalidad mínima. De manera que se asocian tantas propiedades, como indique la restricción, al concepto instanciado. Estas propiedades reciben como objeto un valor por defecto, si la propiedad padre es de tipo, o bien un concepto por defecto, si la propiedad padre es de concepto.
- Cuando la primitiva gráfica está asociada a una propiedad, la creación de una nueva figura implica la instanciación de una propiedad en el esquema asociado a la vista gráfica que contiene la primitiva. Cuando se deja la primitiva sobre la hoja de figuras, se pide la información sobre los conceptos que va a relacionar dicha propiedad. Esta información permite asociar la figura que representa gráficamente la propiedad con las figuras que representan los dos conceptos que relaciona. Las

propiedades que describen la propiedad padre también son instanciadas como se ha indicado para un concepto.

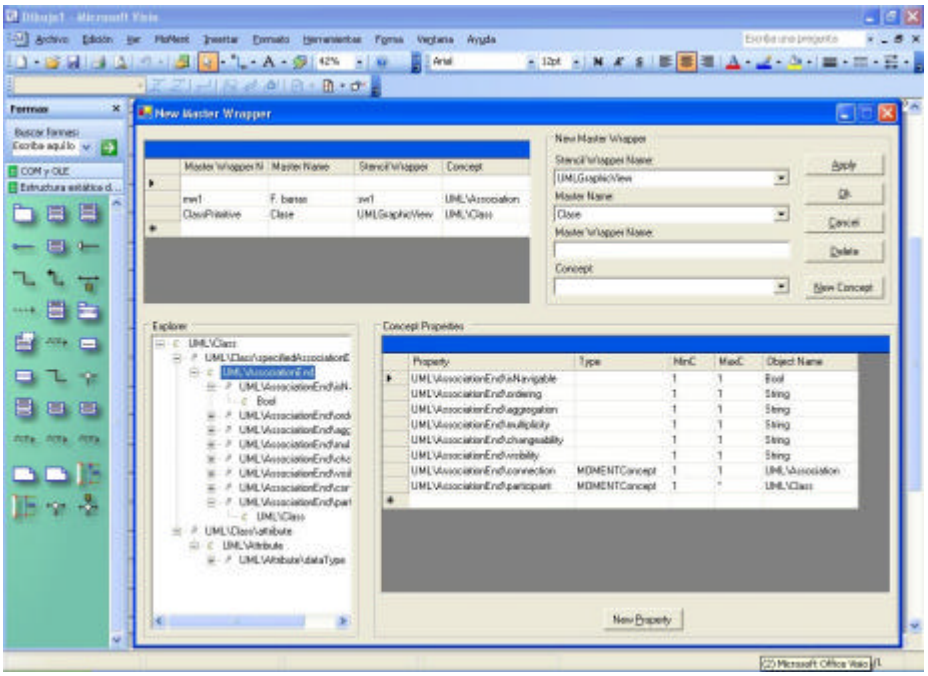


Fig. 5. Creación de los conceptos y propiedades de un metamodelo

Una vez creada la figura sobre la hoja de figuras se puede consultar la especificación algebraica que define el concepto o la propiedad de manera visual, de una forma similar a cómo se realiza en el formulario de la Fig. 5. Además se permiten instanciar aquellas propiedades que no tienen definida ninguna restricción sobre la cardinalidad mínima. De manera que no sólo se define un modelo gráfico, sino que además se utiliza la información semántica asociada a un metamodelo, tanto para los conceptos como para las propiedades.

La especificación algebraica subyacente a un modelo gráfico se almacena en un repositorio RDF a través de la plataforma MOMENT, mientras que la hoja de figuras se almacena en un documento con el formato específico que utilice la herramienta CASE. En el caso de VISIO, el documento que almacena un modelo gráfico tiene extensión *.vsd*.

Además, cuando disponemos de un metamodelo en MOMENT que ya ha sido asociado a una metáfora gráfica, se puede cargar un modelo de ese metamodelo, directamente del nivel M1 de MOMENT. Automáticamente se asocian las figuras, que son instancias de las primitivas gráficas de la vista asociada al metamodelo, a los términos concepto y propiedad del modelo formal, dando lugar a un diagrama que representa gráficamente el modelo cargado mediante la metáfora del metamodelo.

5. Trabajos relacionados

La definición de metáforas gráficas para técnicas de modelado no es una idea nueva en Ingeniería del Software como se indica en [8], pero pocas son las herramientas que permiten especificaciones formales, por no decir ninguna de las herramientas CASE más conocidas. En esta sección tratamos las herramientas CASE más conocidas en dos ámbitos de modelado: el independiente de dominio y el específico a dominio.

La técnica de modelado más representativa de las independientes de dominio es, sin duda alguna, UML [2]. UML aporta una notación gráfica para diseñar productos software siguiendo el paradigma orientado a objetos, pero no proporciona una especificación formal y sólida que garantice la producción de software de calidad y consistente.

La falta de una definición formal de los elementos del estándar UML provoca que el propio estándar vaya evolucionando sin garantía alguna sobre compatibilidad entre las diferentes versiones. De esta manera, una herramienta CASE que soporte UML, perfectamente puede quedar desfasada tras la publicación de una nueva versión del estándar. La existencia de una definición formal permitiría utilizar herramientas para demostrar que la nueva versión es compatible con la anterior o incluso facilitar la migración automática hacia la nueva versión.

No obstante, hoy en día, existen herramientas CASE que permiten generar un producto software total o parcial a partir de un modelo UML, como Rational Rose [9], MS Visio [10], ArgoUML [11]. Estas herramientas únicamente suelen generar el esqueleto de la aplicación software final y las herramientas que proporcionan productos finales suelen ser difícilmente extensibles, lo que no permite seguir la contiua evolución de UML, y se suelen centrar en la generación de un tipo específico de aplicaciones. La funcionalidad de algunas de estas herramientas ha sido embebida en entornos de desarrollo (como Rational Rose XDE Developer [12] para Visual Studio .NET y EMF (Eclipse Modeling Framework) [13] para Eclipse) explotando los modelos UML para proporcionar facilidades durante la implementación del producto software. En [14] se realiza un amplio estudio de las propuestas de formalización de UML, pero ninguna de ellas ha llegado a sensibilizar OMG para enriquecer el estándar UML.

Por otra parte, las técnicas de modelado específicas a dominio se están consolidando como una fuerte alternativa a las independientes de dominio. Esto es debido a que permiten mejorar el proceso de producción de software especializando a los usuarios en el espacio del problema (su área de trabajo) y no en el espacio de la solución (implementación tecnológica). La computación integrada en modelos (MIC – Model Integrated Computing) [15] es una aproximación de desarrollo de sistemas y de software que permite el uso de modelos específicos a dominio para representar aspectos relevantes de un sistema. El ciclo de desarrollo MIC consiste en identificar los conceptos de un dominio, sus atributos y relaciones obteniendo un metamodelo concreto. Este metamodelo se traduce a un entorno de diseño específico del dominio (DSDE – Domain Specific Design Environment) que permite definir modelos en un dominio. Además los DSDE suelen obtener código ejecutable, realizar análisis o animar los modelos. Algunas de las herramientas que destacan en este campo son GME [16], Atom3 [17] y MetaEdit+ [18]. Todas estas herramientas constituyen un

marco de metamodelado basadas en un lenguaje abstracto para definir la sintaxis, semántica y visualización de lenguajes específicos a dominio.

En cambio, muchas de ellas utilizan lenguajes abstractos propietarios o UML (como GME) para definir metamodelos en lugar de utilizar un formalismo adecuado. De esta manera, los intérpretes de modelos obtenidos a partir de un metamodelo concreto suelen tener una complejidad considerable. Además, los transformadores de modelos se basan en las capacidades de estos lenguajes para realizar dichas traducciones de modelos a metamodelos diferentes. Por otra parte, ninguna de ellas consta de un mecanismo para definir metáforas gráficas como el que proporciona el entorno gráfico de MS Visio.

Mediante el *add-in* que hemos desarrollado para dicha herramienta CASE, no somos sólo capaces de definir metamodelos específicos a dominio de una forma sencilla y consiguiendo una buena representación gráfica de los conceptos relevantes de un dominio, sino que también ofrecemos soporte formal para notaciones gráficas utilizadas masivamente para el modelado independiente de dominio, como UML. Esta aportación se consigue mediante las especificaciones algebraicas que representan artefactos software en MOMENT.

6. Conclusiones y trabajo futuro

Las técnicas formales de modelado aportan propiedades como la precisión y la corrección, y permiten dar soporte automatizado al proceso de desarrollo de software. Sin embargo, la complejidad de estas técnicas suele provocar rechazo a la hora de utilizarlas en la práctica. Mediante este trabajo mostramos un mecanismo que salva este inconveniente representando gráficamente los conceptos tratados mediante los formalismos, permitiendo un uso intuitivo y sencillo de especificaciones algebraicas. De manera que la asociación de una ontología visual a una determinada ontología formal permite explotar las propiedades de las herramientas formales desde un entorno de modelado visual.

En este artículo se presenta un mecanismo de representación gráfica para los metamodelos especificados algebraicamente en MOMENT. De esta manera, se permite definir metáforas gráficas asociándolas a metamodelos formales utilizando una herramienta CASE de gran difusión y fácilmente extensible. Este hecho ofrece MOMENT como un entorno metodológico que permite asistir el desarrollo de software automatizando procesos de transformación de modelos, utilizando siempre una interfaz visual que permita una interacción sencilla con el usuario.

Para llegar a tal fin, hemos descrito la plataforma MOMENT y los principales elementos del entorno gráfico de una herramienta CASE. A continuación hemos descrito la arquitectura de un módulo que permite extender dichas herramientas, las asociaciones entre sus elementos gráficos y los elementos de la plataforma MOMENT, contextualizando la implementación en la herramienta Visio 2003. Finalmente, detallamos la interfaz que permite especificar dichas asociaciones para un metamodelo específico.

Mediante la aproximación seguida, MOMENT se beneficia del entorno gráfico de herramientas CASE, evitando el coste de desarrollo que sería necesario invertir para

obtener un editor gráfico de las mismas características. Siguiendo este enfoque, se está estudiando la posibilidad de desarrollar un plug-in para la herramienta Eclipse que se beneficie de su módulo Graphical Editor Framework [20] y que utilice MAUDE [21] como sistema de reescritura de términos para manipular artefactos software en MOMENT.

Por el momento, la metáfora gráfica que se asocia a un metamodelo y el propio metamodelo formal se almacenan y cargan por separado. Como siguiente paso, vamos a resolver este inconveniente fusionando ambos en un nuevo tipo de proyecto (*add-on*), de manera que cuando se elija definir un modelo de ese tipo directamente se cargará toda la información gráfica y semántica necesaria.

7. Bibliografía

1. Boronat A., Carsí J.A., Ramos I., *Una plataforma semántica para la gestión de modelos*, Jornadas de Ingeniería del Software y Bases de Datos, JISBD 2003. Alicante, 2003.
2. Object Management Group. *Unified Modeling Language (UML) 1.4 draft*, February 2001. http://www.omg.org/cgi-bin/doc?ad/2001-02-11
3. Clavel M., Durán F., Eker S., Lincoln P., Martí-Oliet N., Meseguer J. and Quesada J., *Maude: Specification and programming in rewriting logic.* Theoretical Computer Science, 2001.
4. Graham Wideman, *Microsoft Visio 2003 Developer's Survival Pack*, Ed. Trafford, 2004.
5. F# web site: http://research.microsoft.com/projects/ilx/fsharp.aspx
6. Diaconescu, R., Futatsugi, K., Ishisone, M., Nakagawa, A. T. and Sawada, T. *An Overview of CafeObj*. In: C. Kirchner and H. Kirchner (eds), Proceedings of WRLA '98, Electronic Notes in Theoretical Computer Science, Vol. 13, 1998.
7. Object Management Group, *Meta-Object Facility (MOF) version 1.4*, April 2003. http://www.omg.org/technology/documents/formal/mof.htm
8. Bruno G., "Model Based Software Engineering", Chapman & Hall, 1995.
9. Rational Rose Developer. Official web site: http://www-306.ibm.com/software/rational/
10. MS Office Visio 2003. Official web site: www.microsoft.com/office/visio/
11. ArgoUML. Official web site: http://argouml.tigris.org/
12. Rational Rose XDE Developer. Official web site:
http://www-306.ibm.com/software/awdtools/developer/rosexde/
13. Eclipse Modeling Framework. Official web site: http://www.eclipse.org/emf/
14. J.L. Fernández Alemán, "Una Propuesta de Formalización de la Arquitectura en Cuatro Capas de UML", Tesis Doctoral, Departamento de Informática y Sistemas, Universidad de Murcia, 2001.
15. Sztipanovits J. and Karsai G., *Model-Integrated Computing*, Computer, Apr. 1997, pp. 90-92.
16. A. Ledeczi., M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle, and P. Volgyesi. *The Generic Modeling Environment*. In Proc. Workshop on Intelligent Signal Processing. 2001.
17. Lara J., Vangheluwe H., Alfonseca M., *Using Meta-Modelling and Graph Grammars to create Modelling Environments*, Workshop on Graph Transformations and Visual Modellig Techniques GT-VMT at ICGT'2002. Barcelona, October 2002. To appear as Electronic Notes in Theoretical Computer Science Vol.72, (3).
18. Kelly, S., Lyytinen, K., and Rossi, M., *Meta Edit+: A Fully configurable Multi-User and Multi-Tool CASE Environment*, In Proceedings of CAiSE'96, Lecture Notes in Computer Science 1080, Springer-Verlag, Heraklion, Crete, Greece, May 1996, pp. 1-21.

19. D. Beckett. The Design and Implementation of the Redland RDF Application Framework. Tenth International World Wide Web Conference. May 2-5, 2001, Hong Kong. http://www10.org/cdrom/papers/490/
20. Graphical Editing Framework. Official web site: http://www.eclipse.org/gef
21. MAUDE. Official web site: http://maude.cs.uiuc.edu/
22. Boronat A. Carsí J. A., Ramos I., Una *Plataforma para la Gestión Formal de Modelos*, Informe técnico. Ref. No: DSIC-II/4/04. Pages:11. Julio 2004.
23. Object Management Group, *Model Driven Architecture (MDA)*, ormsc/01-07-01, July 2001. Available at http://www.omg.org/cgi-bin/doc?ormsc/01-07-01

Usability Metrics in Adaptive Agent-Based e-Learning Systems

Víctor López-Jaquero, Antonio Fernández-Caballero and Pascual González

Laboratory of User Interaction and Software Engineering
Computer Science Research Institute
University of Castilla-La Mancha, Albacete (Spain)
{victor, caballer, pgonzalez}@info-ab.uclm.es

Abstract. Human-computer interaction in traditional application development is focused on the interaction between tasks and a single user interface designed for a single kind of user. A logical evolution should lead interaction to a development model where user skills and preferences are taken into account. In this paper, we introduce preference metrics and performance metrics as parameters that enable user interface adaptation in tutoring systems. The proposal includes a practical example for learning/teaching of an engineering course.

1 Introduction

The ultimate goal for Human-Computer Interaction (HCI) must be the creation of user interfaces based on each individual user preferences (López-Jaquero, Montero, Fernández-Caballero & Lozano, 2003). Those preferences can be captured initially, to a certain extent, in analysis development stages. Using those captured data user profiles can be created in concordance with the identified user stereotypes. However, the user advances in his knowledge, and his preferences change. We need to understand the way the user "uses" the application, and that is where usability metrics can do the job for us.

We introduce in this paper how usability metrics applied to intelligent tutoring systems can lead to the definition of high adaptive learning/teaching systems. These systems will be able to adjust to a specific student and even they will be able to recommend to teachers how to improve the course. Finally, to achieve our goal to create a highly adaptive teaching environment it will take some artificial intelligent techniques that will be modelled by means of multi-agent systems (MAS). An Intelligent Tutoring System is proposed, in which usability metrics are used to achieve intelligent behaviour of the system to improve learning.

2 Usability Metrics

Usability metrics are software quality metrics with a long history of successful application in software engineering (Card &Glass, 1990; Gilb, 1977; Henderson-Sellers, 1996). But, metrics also carry risks (Constantine & Lockwood, 1999). No simple number can completely represent anything as subtle and complex as the usability of a software system, but numbers can sometimes create the illusion of understanding.

Usability metrics have a number of uses, but mostly from the designer's point of view. Metrics for usability can be thought of as falling into three broad categories: preference metrics, which quantify the subjective evaluations and preferences of users, performance metrics, which measure the actual use of working software, and predictive metrics, or design metrics, which assess the quality of designs and prototypes. We shall focus on preference and performance metrics.

One of the most popular ways to assess usability is to use preference metrics. User satisfaction is a component of usability and also an important factor in success in the marketplace. One good example of a standardized set of preference metrics is the Software usability Measurement Inventory (SUMI) developed as part of the ESPRIT project (Porteous, Kirakowski & Corbett, 1994). SUMI is a 50-item questionnaire that includes five subscales measuring different subjective aspects of usability: affect, efficiency, helpfulness, control, and learn ability. Another approach is the Subjective Usability Scales for Software (SUSS) questionnaire, which measures six key elements of user interface designs affecting usability: valence, aesthetics, organization, interpretation, acquisition, and facility. Preference metrics are one of the pillars for user interface customization. However, because of their intrinsic characteristics, they are difficult to assess at run time. There are some preference metrics, such as the manipulation artefact used when commanding tasks (keyboard, menus, and toolbars) that can become especially useful for capturing user preferences.

Performance metrics are indices of various aspects of how users perform during actual or simulated work. Measurement studies form the basis of much traditional research on human factors. User performance is almost always measured by having a group of test users perform a predefined set of test tasks while collecting time and error data (Nielsen, 1993). Typical quantifiable usability measurements include: the time users take to complete a task; the number of tasks of various kinds that can be completed within a given time limit; the ratio between successful interactions and errors; the time spent recovering from errors; the number of user errors; and so on (Nielsen, 1993). Of course, only a subset of these measurements would be collected during any particular study. Performance metrics are especially useful for assessing overall usability. One important point for this kind of metrics is that most of them can be evaluated at run time in a simple manner. Performance metrics are one more input parameter to advance towards user interfaces adapted to the user. Our proposal is to leave behind user interfaces where the user must adapt to a given and fixed interface.

3 An Adaptive Agent-based Tutoring System

The architecture proposed so far is being tested in e-learning system as an Intelligent Tutoring System (ITS) for an Engineering course taught at the Polytechnic Superior School of Albacete, University of Castilla-La Mancha. One of the main goals is that the alumni learn more and better, that is to say, to be able to structure learning matter in such a way to facilitate the learning facilities. One characteristic to take into account in learning is the rhythm the student is able to learn. Thus, an ITS has to adapt rhythm it introduces the concepts to the learning rhythm of each student (for instance, to show more or less exercises, to show more or less tests, etc.). Another aspect widely considered in learning theory is reinforcement by rewarding a correct answer and penalizing the errors (by means of messages, sounds, etc.). Another goal in our environment is to enhance teaching as well as learning. One of the main problems a professor faces when teaching is that he does not know the skills of his alumni. Our proposal leads to conclusions that "teach how to teach".

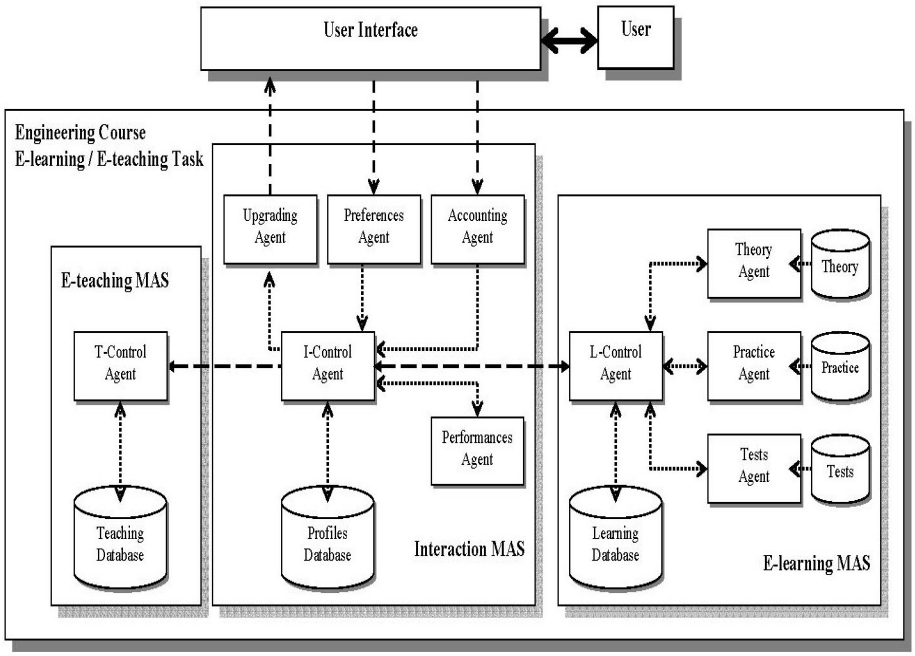


Fig. 1. Adaptive agent-based tutoring system architecture

In our teaching system (see figure 1) there are three multi-agent systems: (1) The Interaction MAS, which takes care of the user. It captures user preferences by means of usability metrics to build profiles. The contents shown to the user will be created according to the preferences and skills captured to improve learning experience. (2) The E-Learning MAS composes contents for the user. Contents are made of three different parts: theory, exercises and tests. All three parts are composed according to

the information captured by the Interaction MAS. (3) The E-Teaching MAS is one of the most important parts according to our experience. It makes recommendations for improving our day-to-day classes for that course.

The user (the student) is in front of the user interface. From the interaction of both entities, modelled by the Interaction MAS, different metrics that are stored in a Profiles Knowledge Database (KDB) are collected. This database contains the different profiles as a result of the use of the system by different students, with different aptitudes, motivations, etc. The multi-agent system for the learning (E-Learning MAS), gets data obtained from the profiles (analysis of the distinct metrics captured) and adequates the contents shown to the concrete student that accesses the Web site. On the other hand, the multi-agent system for teaching (E-Teaching MAS) obtains measures that permit to get recommendations to enhance the course. Finally, to offer a good learning of the course, the latter has been decomposed into theory, exercises and tests.

3.1 E-Learning MAS

The Learning MAS appears from the general goal to maximize the course learning. The learning control agent communicates bi-directionally (asks for and receives information) with the theory agent, the exercises agent, the tests agent and with the interaction control agent (Interaction MAS). This agent asks for/receives Theory Web Pages to/from the theory agent, asks for/receives Exercises Web Pages to/from the exercises agent, asks for/receives Tests Web Pages to/from the tests agent and communicates (through the interaction control agent) with the performance agent to record the performance of the student in order to decide if he needs a reinforcement. If the student needs some kind of reinforcement the learning control agent will elaborate a plan with the material that has to be shown to the student. In order to determine if the student needs reinforcement the performance agent will have access to a KDB where the minimum requisites for each subject are stored (quantity of exercises to be initially shown to the student, how many exercises the student has to answer correctly, and in how much time, maximum time to correctly answer an exercise, etc.).

The theory agent is constantly waiting for the learning control agent to ask for a Theory Web Page. When this occurs, it looks for the proper theory page and sends it to the learning control agent.

The exercises agent is autonomous as it controls its proper actions in some degree. The agent by its own means (pro-active) selects the set of exercises to be proposed in the subject studied by the student and adds to each exercise the links to the theory pages that explain the concepts related to the exercise. It sends to the learning control agent a Web page containing the exercises proposed.

The tests agent is continuously listening to the learning control agent until it is asked for Tests Web Pages. The agent by its own means (pro-active) goes on designing a set of tests for the subject the student is engaged in. These tests will be shown to the student in form of a Web.

3.2 E-Teaching MAS

The Teaching MAS is the result of the second general goal fixed, namely, to maximize the teaching capacity of the course. The Teaching MAS will be collecting the goodness or badness of the parameters defined for the learning system. The Teaching MAS is pro-active in the sense that it will be providing recommendations to the teacher on those parameters.

3.3 Interaction MAS

The Interaction MAS has been conceived to facilitate the adaptive communication between the system and the user. The interaction control agent tells the upgrading agent what the user preferences are, as obtained by the preference agent, and which has to be the next Web page to be shown (learning control agent of Learning MAS). When speaking about the preferences of the student, we mean the type of letter, the colour, the icons, etc., the user prefers. The information collected is stored in the Profiles KDB. All information concerning time-related parameters and some of the user's behaviours are obtained through the performance agent and the accounting agent.

The preference agent perceives the interaction of the user with the user interface and acts when the user changes his tastes. The preference agent is continually running to know the student's preferences at any time.

The performance agent calculates the performance metrics when the student leaves the system (at the end of a working session) y goes evaluating everything the student does in order to know if he needs reinforcement. It is autonomous and pro-active; as it may calculate metrics at the same time the student performs other tasks. Some of the metrics the performance agent handles are: for each Theory Web Page, the mean time alumni spend there; for each exercise Web page, the mean punctuation obtained by the alumni, as well as the time spent to get the correct answer; for each Tests Web Page, the mean time spent to answer all questions, and the mean punctuation obtained in the tests.

The accounting agent perceives the interaction between the student and the user interface and acts (gets information) when the student changes to another Web page, scrolls up and/or down, performs an exercise or a test, etc.

Finally, the upgrading agent is constantly waiting for the interaction control agent to ask to update the user interface with the new information to be shown to the student (to show another Web page or to show the same Web page but changed to the new tastes of the student).

4 Conclusions

User interface generation has become a software engineering branch of increasing interest. This is probably due to the great amount of money, time and effort spent to develop user interfaces, and the increasing level of exigency of user requirements for

usability and accessibility ("W3C", 2002) compliances. Besides it, users engaged in HCI are becoming more and more heterogeneous, and that is a fact we cannot ignore.

In this paper we have proposed an architecture that considers the high diversity of users' skills and preferences: a user-centred and adaptive interaction multi-agent system. This architecture is inspired in usability metrics.

Acknowledgement

This work is supported in part by the Spanish Junta de Comunidades de Castilla-La Mancha PBC-03-003 grant.

References

Card, D., & Glass, R. (1990). Measuring Software Design Quality. Prentice-Hall.

Constantine.L.L., & Lockwood L.A.D. (1999). Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design. Addison-Wesley.

Gilb,T. (1977). Software Metrics. Winthrop Publishers, Inc., Cambridge MA.

Henderson-Sellers, B. (1996). Object-Oriented Metrics: Measures of Complexity. Prentice-Hall.

López-Jaquero, V., Montero, F., Fernández-Caballero, A., & Lozano, M.D. (2003). Towards adaptive user interfaces generation: One step closer to people. Proceedings of International Conference on Enterprise Information Systems 2003.

Nielsen, J. (1993). Usability Engineering. Academic Press.

Porteous, M., Kirakowski, J., & Corbett, M. (1993). SUMI User Handbook. University College Cork, Ireland.

W3C. (2002). http://www.w3.org/WAI/

A Controlled Experiment for Measuring the Usability of WebApps Using Patterns

Francisco J. García, María Lozano, Francisco Montero,
José A. Gallud and Carlota Lorenzo

Universidad de Castilla – La Mancha. Avda. de España s/n
0271 - Albacete – Spain
fcoj.garcia3@alu.uclm.es
{mlozano, fmontero, jgallud}@info-ab.uclm.es
carlota.lorenzo@uclm.es

Abstract. Usability has become a critical quality factor of software systems in general, and especially regarding Web-based applications. Measuring quality is the key to developing high quality software, and it is widely recognized that quality assurance of software products must be assessed focusing on the early stages of the development process. This paper describes a controlled experiment carried out in order to corroborate whether the patterns associated to a quality model are closely related to the final Web application quality. The experiment is based on the definition of a quality model and the patterns associated to its quality criteria to prove that the applications developed using these patterns improve its usability in comparison with other ones developed without using them. The results of this experiment demonstrate that the use of these patterns really improves the quality of the final Web application in a high degree. The experiment is formally based on the recommendations of the ISO 9126-4.

1 Introduction

Usability has become a critical quality factor of software systems in general, but with the increase use of internet for everyday activities, it is especially important in web-based applications. Usability is a key factor for users to decide whether or not a web application (WebApp) is satisfying.

Thus it is important to design WebApps with a basic level of quality in general and usability in particular, and also developing methods which allow evaluating this quality. In this sense different Web metrics have been defined (Ivory, 2001), (ISO, 2001), which can help us to measure the final usability or quality in use of a web application.

Lots of efforts have been made in order to improve the quality of software systems, such as definition of quality metrics (Olsina, 1999), usability metrics (Ivory, 2001), usability evaluation methods (Nielsen, 93), (Constantine et al., 2000), etc. All these

mechanisms allow us to check the usability of a software system but they have to be used on a final and running application. Nowadays, it is widely recognized that quality assurance of software products must be assessed focusing on the early stages of the development process. It is necessary to incorporate new mechanisms at the very beginning of the software process to produce high-quality software applications. In this sense, the use of design patterns in general has proved to be good for these purposes, especially interaction patterns regarding usability.

Some interaction patterns have been proposed in the literature, but the novelty of our approach is to establish a clear association between a quality factor defined in a quality model and one or more concrete patterns in such a way that the use of that pattern in the construction of a WebApp makes it to assess the corresponding quality criteria.

To demonstrate the validity of these associations and the goodness of using interaction patterns to satisfy the corresponding quality factors, a controlled experiment has been carried out.

This paper is organized as follows: firstly we show a general idea about web quality, usability and patterns, to have a better vision of the work frame where this study is included and the related work.

Then we describe the experiment that we have carried out to prove the utility of patterns and quality models to design quality WebApps as well as metrics as mechanism for evaluating usability.

Finally, we finish the paper with interesting conclusions based on the experiment results and we propose future lines of investigation.

2 Related Work

Before describing the experiment carried out for this study is necessary to talk about web quality and usability to establish the aim and the context of the experiment.

Brajnik states (Brajnik, 2002) that the quality in this context is a property of a website defined in terms of a system of attributes, like consistency of background colours or average download time. ISO 9126 defines web quality dividing it in more abstract terms, as effectiveness, efficiency and satisfaction. (ISO, 2001).

Because of the variety and quantity of attributes proposed in the literature is necessary to develop a quality model that helps to know which attributes are important for the analysis, which one is more important than others, and which measurement methods have to be used to assess the attributes values.

Usability is widely recognized as one of the most important attributes regarding Web quality, so we focus on usability to define a complete quality model (see figure 1) and define an accurate association between the final and more concrete criteria with one or more interaction pattern. The experiment aims to prove the goodness of the quality model and the interaction patterns associated.

The model is centred specifically on usability criteria, as regarding web environments this factor is more meaningful than others because of the new interactive features of internet.

This quality model (Montero et al. 2003) centred on usability is based on usability features defined by ISO 9126, some ergonomic criteria and its sub-criteria. These

ergonomic criteria are implemented by means of patterns, as a way to materialize the abstract concept of a quality criterion with something implementable as it is a pattern.

Pattern concept in computer science in this context is defined as a tupla of three elements: a problem, a context and a solution. Many patterns have been proposed for web development (Percel et al.,1999), (Tidwell, 2002), (Welie, 2003), (Van Duyne et al., 2002), (Rossi et al., 12), or (Montero et al., 2002b) can be cited.

Once we have seen the general terms and web quality model we based the experiment on we continue this paper describing the case study about measure the usability of an e-commerce WebApp by using patterns and the complete description of the experiment.

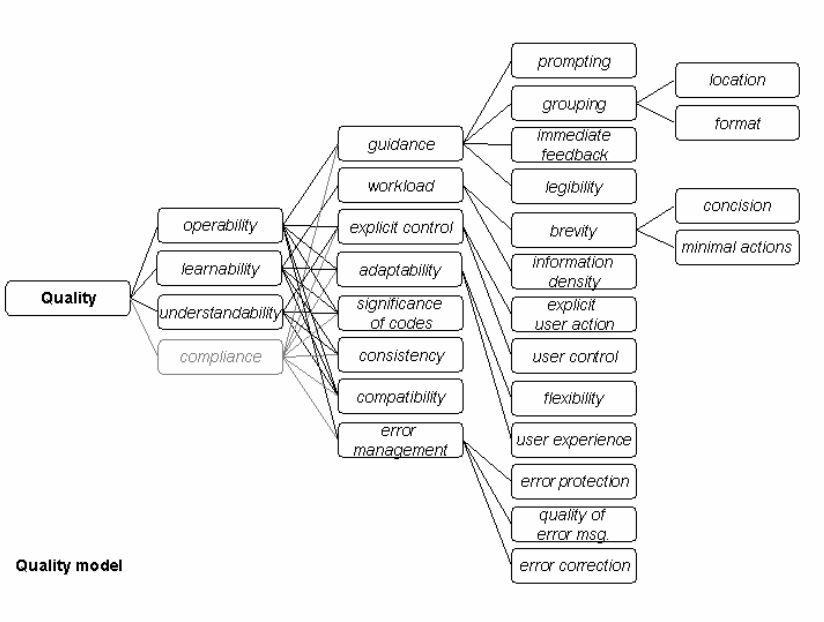


Fig. 1. Web Quality Model

3 Description of the Experiment

The description of the experiment follows the structure and the recommendations of ISO 9126-4.

3.1 Goals

The website that we use for the experiment is a fictitious store named e-fashion in order to eliminate the effects of prior experience. This site is an e-commerce WebApp where you can buy men and women clothes.

The content of online shop is based on a homepage (Nielsen, 2001) which includes the same links and websites as other online apparel stores.

We made six different versions of this site according to various quality patterns we wanted to validate.

The main goal of the experiment is:

- To prove that the association between interaction patterns and the criteria defined on the proposed quality model (Montero et al. 2003) is correct (see figure 1), in such a way that the use of the patterns in the construction of WebApp makes it to assess the corresponding quality criteria, and for this reason the WebApp versions that includes the recommended patterns for solving usability problems have more quality that the ones that do not include them. In this case we carried out the experiment choosing only one of the quality model criteria: "Guide". The same method could be used with the rest of quality factors and patterns related.

We had to evaluate the websites designed for the experiment using usability metrics to validate the hypotheses.

3.2 Method

To do the experiment for each feature of the criteria "Guide" represented in the quality model we based on, we had to design six versions of e-fashion. Thus we had to choose six different groups of people, with the same experience on using internet and buying in electronic shops.

Each group had to do four tasks, and then we had to measure them with usability metrics to reach the goals of the experiment.

Each version of e-fashion was made using the patterns that are recommended according to the features of the quality criteria "Guide" of our quality model (Montero et al. 2003). We wanted to measure if the association established between the patters and this features is correct and their use improves the usability of WebApp generated.

The sub-criteria defined for the criteria "Guide" are the following:

-Prompting
-Grouping
-Immediate feedback
-Legibility

The patterns, defined by D. Van Duyne (Van Duyne et al. 2002), associated to each sub-criteria are:

- Patterns associated with the sub-criteria Prompting:

D3*: Headlines and blurbs*
D9*: Distinctive HTML titles*
G1*: Featured products*
G2*: Cross-selling and up-selling*
H6*: Pop-up windows*

H8*: Context-sensitive help*
K6*: Location bread crumbs*

- Patterns associated with the sub-criteria Grouping:

B3*: Hierarchical organization*
B4*: Task-based organization*
B5*: Alphabetical organization*
B6*: Chronological organization*
B7*: Popularity-based organization*
D1*: Page templates*
D7*: Inverse-pyramid writing style*
F1*: Quick-flow checkout*
G1*: Featured products*
H7*: Frequently asked questions*
J3*: Organized search results*
K1*: Navigation bar*

- Patterns associated with the sub-criteria Immediate feedback:

C2*: Up-front value proposition*
D9*: Distinctive HTML titles*
F3*: Shopping cart*
F7*: Order summary*
F8*: Order confirmation and thank-you*
H6*: Pop-up windows*
I3*: Clear first reads*
K5*: High-visibility action buttons*
K10*: Obvious links*
K14*: Page not found*

- Patterns associated with the sub-criteria Legibility:

D7: *Inverse-pyramid writing style*
D9*: Distinctive HTML titles*
I2*: Above the fold*
I3*: Clear first reads*
I4*: Expanding width screen size*
I5*: Fixed-width screen size*

Taking into account these associations, we implemented six different websites of e-fashion: one that uses all the patterns, called e-fashion 3 another that do not use them, e-fashion 4, another that uses the patterns defined for the sub-criteria Prompting, e-fashion 5, and so on to e-fashion 8 that was designed according to the patterns defined for the sub-criteria Legibility.

3.2.1 Participants

The selection of participants was not easy as they had not to be experts on using internet and buying on websites.

Finally the users selected to carry out the experiment were high school students between 16 and 17 years. All of them had the same experience on using internet and no experience on buying on internet. There were only 2 users that had been bought something on internet but just once.

3.2.2 Context of Product Use in the Experiment

3.2.2.1 Tasks

The proposed tasks that each user had to do on the experiment were designed according to the factor "Guide" that we wanted to evaluate and considering that the WebApps designed for the experiment were e-commerce sites.

The tasks that each user had to do were:

- **Task 1**: To buy two grey men jerseys.
- **Task 2**: To write the price of a white woman shirt with black stripes
- **Task 3**: To add to the shopping chart four pairs of uncovered woman shoes and a white man belt.
- **Task 4**: To buy a brown woman skirt.

After the realisation of these tasks the users had to do the satisfaction test proposed in the experiment.

With the results obtained using the metrics we got conclusions about the use of patterns to create usable websites, as described afterwards.

3.2.2.2 Context Used for the Experiment

The evaluation was made on the Albacete high school called CEDES, in Spain, on June the 15th and 16th of 2004.
The participants were observed by the person in charge of the experiment along its duration.

3.2.2.3 Participant´s Computing Environment

All the participants used the same computer machines and the same internet connection. The computers used were Pentium MMX with 32 MB of RAM, with 15" monitors and a screen resolution of 800x600. The operating system was Windows 2000.

The internet connection was 150 kb/s ADSL, but shared by all the machines. This was determining in the development of the experiment, because the loading of the pages was very slow, because of the high quantity of images shown in the web.

3.2.3 Design of the Experiment

Six groups of participants were made, with a media of 12 users per group. Thus the total amount of participants was 74 people. Each group of users did the proposed tasks in one of the versions of e-fashion. Each user had to do the 4 proposed tasks and to fill the satisfaction test.

3.2.3.1 Procedure

When the participants arrived at the laboratories where the experiment was carried out, all of them were informed about the goals of the experiment at the same time. We told them that the experiment was made to measure the usability of the website e-fashion where they had to do the proposed tasks to find out whether it met the needs of users as themselves. They were told to read each task (described on a link on the home page of e-fashion) and to make these task one by one. Finally they had to fill the satisfaction test available by a link on the home page.

The experiment was about 50 minutes long for each one of the six groups. In this time the users had to be able to execute the four proposed tasks and the satisfaction test.

The participants were given basic instructions describing the environment. The evaluator reset the state of the computers before the coming of each new users group, and gave them the appropriate and convenient instructions. The participants were informed too that they were not be helped by the evaluator, because the web was enough to help the users to make the tasks properly.

Any incident occurred during the experiment was solved by the evaluator in the most properly way.

The evaluator finally asked the participants about the difficulties they had encountered to have a best vision about the results of the experiment.

3.2.4 Metrics

The metrics we utilized on the experiment were:

- Efficiency

We use as efficiency metric the "*Task Time"*, which is the time that each user spends completing each task. We did the mean time for all users on each task.

- Effectiveness

The effectiveness was measured using the metric "Task Completion" and "Error frequency". With these metrics we can obtain the number of tasks that users did not completed and the errors that the users had done on each task.

In this case we considered as completed task the one where the user had done what we ask to do, independently of he did it properly or not.

As well as the number of completed task and not completed we show the results as a percentage to allow a simple analysis.

One task is not completed for example if we asked to buy something and the user has added the products to the shopping chart but he does not pay for them.

The results of the metric "Error Frequency" allow us to evaluate the errors that each user has done. If one of the task was to buy a brown skirt, but the user have bought another token, it will be considered that the task is completed but with errors.

If the task consisted on taking note of the price of a certain shirt and this price was written from other shirt, then the task will be considered completed but with errors.

Another possible way to have contemplated this metrics would be to consider that the task with errors it is not completed.

- Satisfaction

The satisfaction metric was measured using a satisfaction test, that was created basing on some questions of the evaluation test SUSS, developed by Constantine (Constantine et al., 1999) to measure key elements in the interfaces design as could be personal tastes, aesthetic, organization, understandability, and learning.

Other questions of the test are based on SUMI test, accepted by expert evaluators and international organizations as ISO.

Our test is about 20 questions that allow us to evaluate in a simple way the satisfaction of the users who have carried out the four tasks in the website.

Each user had to answer each question of the test choosing an option between 1 and 5, like a Likert scale, except in the two first questions and the three final questions that were questions with a scale of three points.

- Compressibility and learning

Finally the metric learning was measured. This kind of metric measures how the user have learned to use the website. To measure this metric we use a special question of the satisfaction test and we compared too the "Task Time" of task 1 and task 4, because these two tasks consisted on buying something. Thus, it was supposed that if the user had been able to learn, the Task Time for Task 4 would be inferior to Task Time for Task 1, as indeed occurred.

3.2.5 Results

Before comparing the data obtained by the metrics in the different versions of e-fashion we can say that the association made between the quality patterns and the feature Guide in general is good and improves notably the quality of websites.

Special case was e-fashion 5 because the server fell down for a minutes and so some tasks could not be completed , and some task time were higher than expected, so the results of e-fashion 5 must be taken carefully taking into account this unexpected situation.

3.2.5.1 Comparative of Medium Task Time

Figure 2 shows a graphic where the medium times of task are represented for each version of e-fashion.

We highlight the fact that it occurs something curious on the versions of e-fashion that use all the patterns of the criteria "Guide", or some of them. These sites had more information to load, for example more quantity of images, than e-fashion 4, the "worst" version of all. For all that the loading time of these pages was slower than the loading time on e-fashion 4.

This workload that made the loading slower was due to products images, for example on the home page that loads the new products of the month, and this could affect to the final time task.

Despite this situation, on the Figure 2 we can see that in general the users take more time to execute the tasks on e-fashion 4 than on e-fashion 3, as we expected. If we add the handicap that has been talked about, we have to take in count the times in function of the load of the page. In this case there is no doubt that e-fashion 4 is the worst of all versions of the sites, and e-fashion 3 is the best of the six versions because its task times are slower on 3 of the 4 tasks.

These results allows us to conclude that the association between the patterns with the criteria "Guide" is good in general and helps to improve the quality of websites.

Each colour of the Figure represents each version of e-fashion that was used on the experiment. The X axis shows the number of task and the Y axis shows the time expressed in minutes.

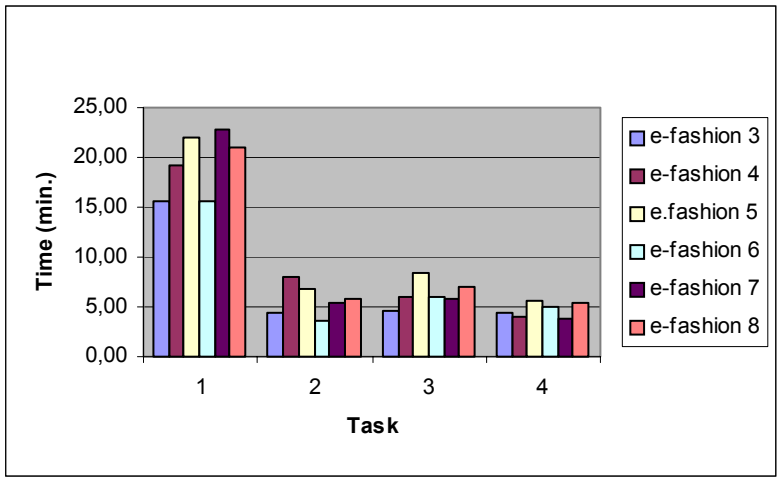


Fig. 2. Medium Task Time Comparative

3.2.5.2 Comparative of Task Completion

Figure 3 shows a graphic where the completion of task is compared on the six versions of e-fashion. We want to remind that when the experiment was made with e-fashion 5 the server fell down, and some tasks had not been completed. So, this 23% of users that had not completed the tasks on e-fashion 5 do not reflect the real usability of the web.

If we analyse the rest of groups, we can see that on e-fashion 4 there was a 90% of completed tasks, what shows that this web was the worst of the six versions, because in e-fashion 7 there were a 2 % of users that do not complete all the task, but is a

better result than e-fashion 4 and in the rest versions there was a 100% of task completed.

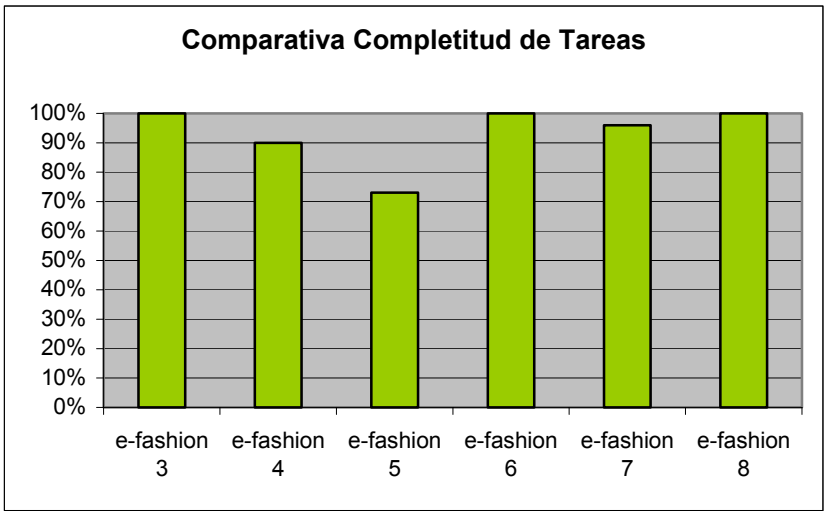


Fig. 3. Task Completion Comparative

3.2.5.3 Errors Frequency comparative

Figure 4 shows which version of e-fashion had more error frequency on its tasks, and as we expected e-fashion 4 was the worst site of the six.

This result reinforces the hypothesis that it is better to design websites according to a quality model and using the ergonomic patterns identified.

3.2.5.4 Satisfaction comparative

Now we show some of the results obtained from the satisfaction test bringing face to face the different versions of e-fashion showing some comparative graphics.

We start with the affirmation "Working with this site is satisfactory", because this question shows the general satisfaction of the users that have used the web application.

On the Figure 5 the reader can see that in general the satisfaction of the users was higher on e-fashion 3, because the most chosen option was number 5, which is the best of the different options. This shows that in general the satisfaction of the users that executed the tasks of the experiment was good.

However on e-fashion 4 the most chosen option was number 2, which is near to the worst option (Totally disagree), which indicates that in general on e-fashion 4 the satisfaction was the worst.

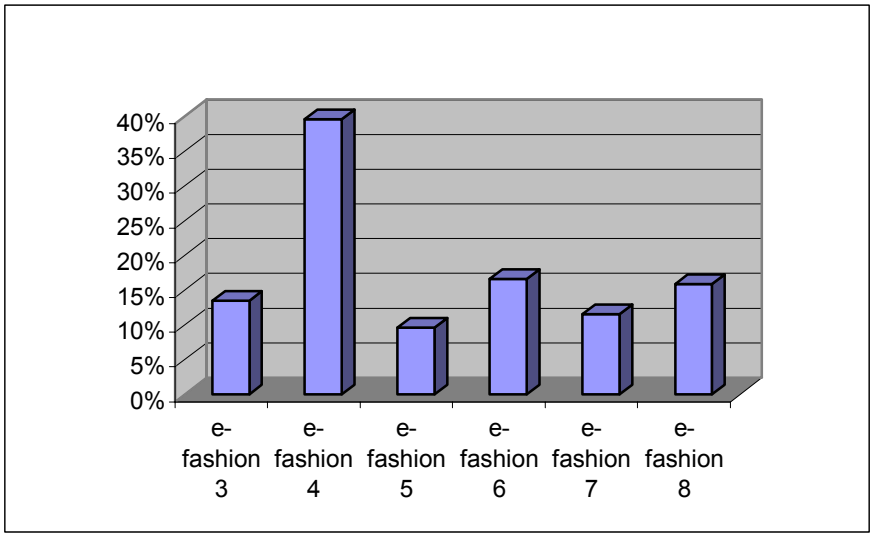


Fig. 4. Errors Frequency Comparative

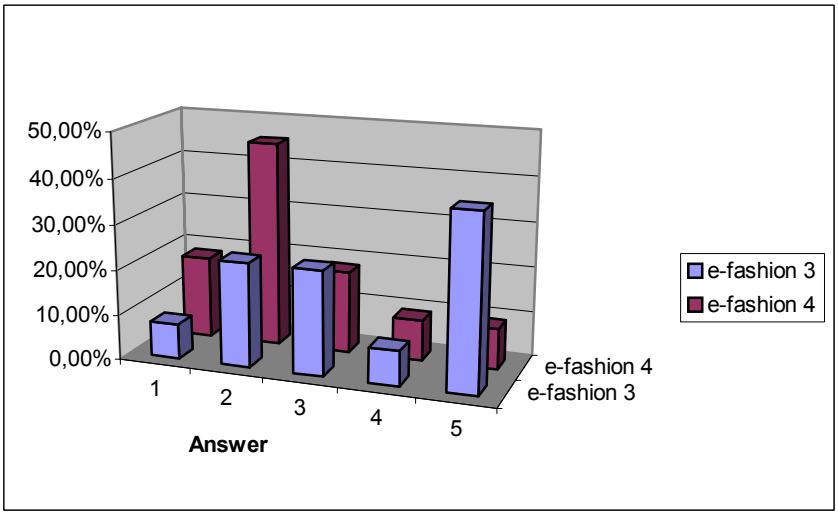


Fig. 5. Answers given by the users to the question "Working on this site is satisfactory"

Another interesting question of the test was the number 4: "The website is very attractive for me", because it shows the capacity of the website to attract users. In this case the results were favourable too to e-fashion 3.

In general the results of e-fashion 3 were positives, if we have on count the percents of the different versions. E-fashion 4 again was the worst of the six versions.

The conclusions that we obtained about the rest of questions in the test were favourable to e-fashion 3 too. This indicates again that using patterns for designing websites is good to improve their final quality.

3.2.5.5 Learning comparative

In this case as we could see in the answers of the question number 21 of the satisfaction test, in the notes taken by the evaluator while the experiment was carried out, and comparing the times obtained on the task where the users had to buy something (number 1 and number 4), the users proved to have understood the functionality of the website and to have learned the basic use of it. Again this result was clearer on e-fashion 3, the version that was implemented using all the patterns defined for the criteria "Guide".

4 Conclusions and Future Works

Based on the interesting results obtained from the experiment, we can conclude that the use of ergonomic patterns to characterize quality criteria defined in a quality model produces better web applications with much more quality than others implemented without using the patterns.

Moreover, the realization of the experiment has been useful to evaluate the metrics used on it. The metrics are useful mechanism for evaluating usability of websites.

With the data obtained from the experiment we can conclude that the satisfaction metric is useful because it shows a good view about what users think when they use the website on true conditions. However we can also state that due to the subjective nature of this metric, and because we cannot assure that all the users says the truth when they fill a test, is important not to use only this metric to evaluate the quality of a website. In this sense, it is better to compare its results with the results obtained by using other metrics, as for instance, Task Time, Error Frequency and so on.

The validity of the experiment is clear in this case as the global results of the metrics used are consistent one with the others.

Another interesting conclusion is that "Task Time" and "Error Frequency" metrics are useful when we need to evaluate the quality of a website, because they show in an objective way the usability of a site when users execute tasks on it. However, in the experiment we can see that it is very important when we want to evaluate the results of the metric "Task Time" the loading time of the page, because the loading time can give a false view about the time users spend on doing the tasks. Thus, it is very important that all the users do the tasks on computers with the same technical specifications.

About the "learning" metric we must say that is a good metric too because it gives a general idea about the capacity of the website to be learned and used by the user. So, applications easier to learn are more satisfactory for the user as we can conclude from the experiment results.

The most important conclusion is that the use patterns to characterize the criteria defined in a quality model allow us to construct web applications with a higher degree of quality than others designed without any reference model nor patterns.

As a final remark, we can state that due to the complexity and difficulty of carrying out an experiment with real people it is much better to develop web quality models which allow developing websites with a basic level of usability. To design websites using a quality model avoids making usability experiments because the final quality will be guaranteed.

References

Brajnik, G (2002). Quality models based on automatic webtesting. Automatically evaluating usability of Web Sites, Minneapolis, CHI April 2002.

Constantine, L.L., Lockwood, L.A.D. (1999), Software for use, A practical guide to the models and methods of usage-centered design. ACM Press

ISO/IEC 9126-1. (2001) Software Engineering – Product Quality- Part 1: Quality Model, International Organization for Standardization, Geneva, 2001.

ISO/IEC 9126-4. (2001) Software Engineering – Software Product Quality- Part 4: Quality in use metrics, 2001.

Ivory, M.Y. (2001) An Empirical Foundation for Automated Web Interface Evaluation. Ph D. Thesis, Berkeley University, California.

Montero, F., Lozano, M., González, P., Ramos, I. (2002) Design Websites by Using Patterns. Second Latin American conference on Pattern Languages of Programming. SugarLoafPLoP02. Itaipava. Rio de Janeiro. Brasil. ISBN: 85-87837-07-9. pp. 209-224.

Montero, F.; Lopez-Jaquero, V., Lozano, M.; González, P. (2003) A Quality Model For Testing the Usability of Web Sites. HCII´03.

Nielsen, J. (1993). Usability Engineering. Morgan Kaufmann.

Nielsen, J. (2001). Homepage Usability: 55 websites deconstructed. New Riders, pp. 315

Olsina, L. (1999) Metodología Cuantitativa para la Evaluación y Comparación de la Calidad de Sitios Web. PhD. Thesis, Universidad Nacional de La Plata, Argentina.

Perzel, K., Kane, D. (1999). Usability Patterns for Applications on the World Wide Web. PloP´99.

Shneiderman, B (1998). Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Publishers.

Tidwell, J. (1999) Commond Ground: A pattern language for HCI design. http://www.mit.edu./~jtidwell/interaction_patterns.html

Van Duyne, D.K.; Landay, J. A.; Hong, J. I.; (2002) The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience, Publisher: Addison Wesley, ISBN: 0-201-72149-X

Welie, M. (2003) Interaction Design Patterns. http://www.welie.com/patterns.

Aproximación a la Evaluación de Interfaces de Realidad Virtual

Arturo Simón, José Pascual Molina y Pascual González

LoUISE – Laboratory of User Interaction and Software Engineering
IIIA – Instituto de Investigación en Informática de Albacete
Universidad de Castilla-La Mancha
Escuela Politécnica Superior, Campus Universitario s/n,
02071 Albacete, España
{arturo,jpmolina,pgonzalez}@info-ab.uclm.es

Abstract. Durante los últimos años se ha estado produciendo una reducción del tamaño de los dispositivos de Realidad Virtual, así como su facilidad de uso y un abaratamiento de los mismos. Esto ha provocado que se investigue en aplicaciones futuras donde se usen estos dispositivos de forma masiva, y esto, a su vez, hace plantearse la cuestión de cómo será la interacción en estos nuevos entornos tridimensionales. Pero, a pesar de las muchas investigaciones realizadas sobre este tema, todavía no se han identificado técnicas de interacción que permitan realizar todas las posibles acciones dentro de un mundo virtual de una forma sencilla y eficiente. Por ello, en el laboratorio de Realidad Virtual del grupo de investigación LoUISE (Laboratory of User Interaction and Software Engineering) del I3A (Instituto de Investigación en Informática de Albacete), se han analizado, diseñado e implementado un conjunto de técnicas de selección y manipulación de objetos, y se ha desarrollado un entorno tridimensional de prueba. Tras esto se han elegido un número de participantes no relacionados con la materia que se encargaran de probar dichas técnicas bajo unas situaciones controladas. Aquí se exponen las conclusiones a las que se ha llegado tras la revisión de los resultados del experimento.

1 Introducción

Dado que a lo largo de la historia reciente de la Realidad Virtual se han propuesto varias técnicas para la interacción dentro de mundos virtuales, lo mejor será atender a una clasificación, en este caso, extraída de las propuestas en [8] y [2], por Bowman y Poupyrev

Metáforas no centradas en el usuario:

- Mundo en miniatura

Metáforas centradas en el usuario:

- Metáforas de mano virtual

- Virtual Hand Clásica
- Go-Go Technique
- Go-Go indirectas

- Metáforas de puntero virtual
  - Ray-Casting
  - Aperture Selection
  - Flash light
  - Image plane

Las metáforas centradas en el usuario son las que ocupan este estudio, y se dividen en dos grupos: las que utilizan una metáfora de mano virtual (o de extensión de brazos) y las que utilizan una metáfora de puntero virtual. En las primeras el usuario debe tocar y coger los objetos con la representación virtual de su mano real. Por otro lado, en las técnicas basadas en la metáfora de puntero virtual, el usuario deberá apuntar hacia los objetos que desee seleccionar o manipular.

En concreto, las técnicas implementadas para este estudio son las siguientes:

- **Virtual Hand:** Esta técnica, descrita en [2] y en [7], consiste básicamente en el uso de un cursor 3D, similar al cursor 2D usado para interactuar actualmente. Los movimientos de la mano del usuario serán asociados a la mano virtual por un sistema de posicionamiento ("tracker"). De esta manera se simula la interacción en el mundo real, resultado pues muy intuitiva, pero introduciendo limitaciones, como son problemas a la hora de interactuar con objetos pequeños o muy grandes, y con los lejanos.

- **Go-Go:** Esta técnica, descrita en [6], consiste básicamente en que el usuario tiene la capacidad de "alargar" su brazo virtual más allá de lo que lo hace su brazo real, manteniendo un esquema de manipulación que sigue basado en el mundo real. Por lo que se mantiene la facilidad mover/rotar un objeto a costa de complicar el proceso de selección, el cual deja de ser tan intuitivo. Además de permite alcanzar objetos a mayor distancia, pero no por ello deja de tener límite ese crecimiento (que vendrá dado por la longitud del brazo del usuario). También dará problemas con objetos pequeños y con los que están fuera de alcance.

  Fast Go-Go: Esta variante contempla el caso en el que no exista la región en la que los movimientos de la mano virtual son correspondencias directas de los de la mano real, por lo que el crecimiento del brazo virtual será mayor que en la técnica anterior (aunque seguirá existiendo límite).

  Stretch Go-Go: En esta variante se divide el espacio en tres regiones concéntricas (lo que la hace menos intuitiva): el área central (los movimientos de la mano real se asocian directamente a movimientos de la mano virtual), área más alejada (brazo virtual crecerá a una velocidad constante alcanzándose cualquier objeto sin importar su distancia con respecto al usuario) y área más cercana al usuario (el brazo virtual decrecerá a la misma velocidad constante anterior).

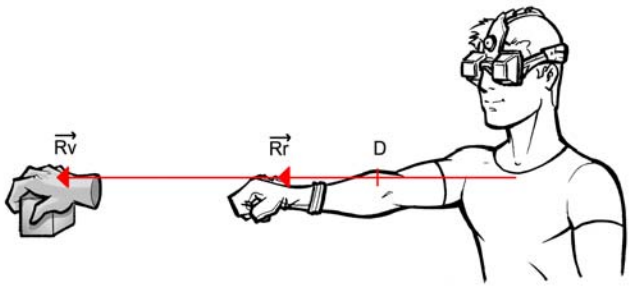


Fig. 1. Crecimiento del brazo virtual en Go-Go Technique (Rr vector real, Rv vector virtual, D distancia a partir de la cual Rv es mayor que Rr, hasta ella son iguales).

- **Ray Casting:** Esta técnica, descrita en [5], [2] y [3], se basa en la idea del puntero 2D de los sistemas de escritorio actuales. Mientras en 2D hay que situar un puntero sobre un icono para seleccionarlo, en 3D se extendería esta idea para usar un rayo virtual infinito que interseque con los objetos para seleccionarlos. Esto ofrece una gran facilidad a la hora de seleccionar cualquier objeto que este a la vista del usuario de una forma relativamente sencilla, exceptuando los pequeños (incluso de cerca, aquí afectara en gran medida el ruido del sistema de posicionamiento). Manipulación limitada (resultará imposible acercarlo o alejarlo de su posición inicial con respecto al usuario) y rotaciones extremadamente complicadas.

  Ray-Casting with reeling: Esta variante soluciona el hecho de que no se pueda cambiar la distancia de un objeto con respecto al usuario, utilizando la metáfora de una caña de pescar ("fishing reel").

- **Spotlight:** Esta técnica, descrita en [4], se diferencia de las anteriores es que utiliza un volumen cónico con vértice en la mano virtual del usuario, en lugar de un rayo. El echo de sustituir el rayo de la técnica Ray-Casting por un volumen cónico se solucionan los problemas a la hora de seleccionar objetos pequeños, pero se mantienen las desventajas, a la vez que se produce ambigüedad más fácilmente, ya que puede colisionar con gran cantidad de objetos cercanos entre si.

2 Descripción del Experimento

2.1 Entorno virtual

Para poder comprobar de primera mano las virtudes y defectos de las técnicas implementadas, se preparó un escenario de prueba. La principal diferencia con estudios anteriores, como los de Bowman[1] o Poupyrev[8], es que en lugar de crear un espacio artificial, donde los objetos están situados en el vacío y donde el usuario

tiene sus cinco sentidos puestos en la técnica de interacción, nuestros entorno de prueba ha sido diseñado cuidadosamente para asemejarse en mayor medida a las aplicaciones existentes de Realidad Virtual, mucho más ricas en gráficos y detalles. De este modo, el interés del usuario no se centraba en la técnica de interacción propiamente dicha, sino en la tarea que requería la aplicación.

Finalmente, el entorno elegido fue una posible oficina de trabajo. Dicha oficina estaba formada por un escritorio, donde se situó objetos al alcance de la mano del usuario, con la idea de que no fuera necesaria ninguna técnica de extensión de brazos para poder alcanzar los objetos situados sobre la mesa. También estaba formado por una estantería, donde estaban situados los objetos fuera del alcance directo del usuario, es decir, donde las técnicas de Virtual Hand Clásica y Go-Go Technique básica no podrían alcanzar. Se incluyeron objetos que sólo lograran "ambientación", o decorativos, que hicieran más creíble el entorno o que, como una ventana, evitaran producir en el usuario una sensación de claustrofobia.

Una vez creado el entorno, se manejaron distintas variables para las tareas de selección. Se varió la distancia entre el usuario y el objeto a seleccionar en dos niveles, los objetos sobre la mesa, y los objetos sobre la estantería. El tamaño se varió en tres niveles, y la cantidad de objetos que rodean al objeto a seleccionar también se varió en dos niveles.

Tabla 1. Variables y niveles considerados.

Variables consideradas	
Distancia del objeto	Sobre la mesa En la estantería
Tamaño del objeto	Pequeño Mediano Grande
Objetos que rodean al objetivo	Sin objetos Con dos objetos

Para cada una de las técnicas se realizó esta batería de pruebas:

- Pruebas realizadas a corta distancia (mesa):
  - C.1.B → Selección de un bolígrafo y su posicionamiento.
  - C.2.F → Selección del flexo localizado sobre la mesa.
  - C.3.C → Selección del cubilete donde están los bolígrafos.

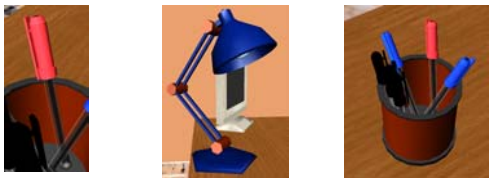


Fig. 2. Objetos a manipular a corta distancia.

- Pruebas realizadas a larga distancia (estantería):
  - L.1.G → Selección de un libro de tamaño grande.
  - L.2.P → Selección de un libro de tamaño pequeño.
  - L.3.C → Seleccionar un cuadro y su posicionamiento.

Fig. 3. Objetos a manipular a larga distancia.

2.2 Participantes

Dada la gran cantidad de pruebas que componían el experimento, pues se contaba con 7 técnicas a evaluar en una gran cantidad de situaciones, se preveía que el tiempo que requerido por cada participante sería bastante elevado, por lo que se decidió que el número de participantes no fuera muy grande. De este modo, se decidió realizar el experimento con 10 sujetos, 5 hombres y 5 mujeres. El propósito de utilizar un número idéntico de participantes de ambos sexos no es otro que el de procurar no condicionar los resultados a un sexo en concreto. Podría incluso afinarse más si se supiera la proporción exacta de hombres y mujeres que suelen utilizar estos sistemas; como no se conocía, se optó por la paridad de sexos.

2.3 Hardware

Para la realización de este estudió se utilizó una estación de trabajo HP Compaq xw6000/PL. Se utilizó el sistema de posicionamiento Ascension Flock of Birds y el ratón 3D Ascension Wanda para que el usuario interaccionara con el sistema. Por último se utilizó un casco de visualización estereoscópica PorView XL 35 para lograr una mayor sensación de inmersión en el usuario.

2.4 Software

La herramienta software elegida para la realización de este estudio ha sido Eon Studio 4.0, de Eon Reality Inc. Eon Studio es una herramienta gráfica de desarrollo de aplicaciones 3D interactivas.

3 Resultados del Experimento

El objetivo de este estudio era el de obtener datos acerca de cada una de las técnicas, con el fin de averiguar, de entre todas las técnicas implementadas, cuál era la mejor en términos de utilidad y usabilidad. Los datos de interés son el *tiempo en completar la tarea* y el *número de errores cometidos*. Para obtener estos datos, había que tener en cuenta el impacto de otras variables en el experimento, que, como ya conocemos, son: la distancia del objeto al usuario, el tamaño del objeto y la técnica de interacción utilizada.

Se pueden identificar como variables independientes en el estudio la distancia y tamaño de los objetos, y la técnica de interacción utilizada. Como variables dependientes de las anteriores el tiempo en completar la tarea y el número de errores o intentos. Por ello el experimento está diseñado para tomar la información de la variables dependientes variando las independientes, de forma que pueda observarse el impacto que éstas tienen tanto en el tiempo como en los errores/intentos a la hora de completar una tarea.

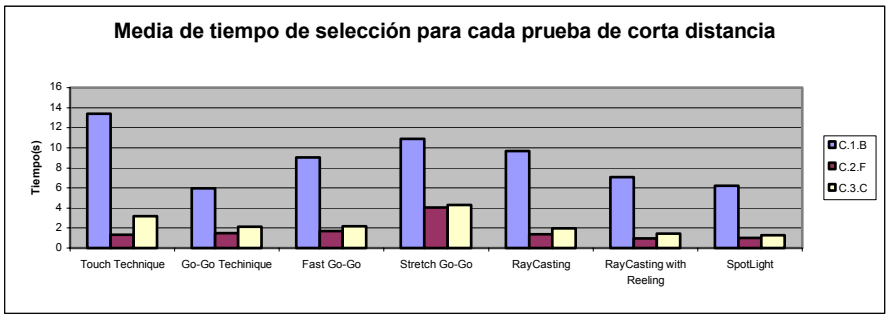


Fig. 4. Media del tiempo de selección para cada una de las pruebas a corta distancia para cada técnica.

Para corta distancia, la prueba que más tiempo lleva realizar es la C.1.B, dado que era la más complicada, puesto que el objetivo era seleccionar un bolígrafo en concreto del cubilete, estando rodeado de otros dos, lo que dificultaba su selección. Por lo demás, y como era de esperar, las técnicas de puntero virtual llevan algo más de tiempo que las de extensión de brazos, dado que es algo más complejo apuntar con el rayo a objetos pequeños, pero es incluso más sencillo cuando los objetos son grandes o medianos (donde todas las técnicas se han comportado más o menos de la misma forma).

Un fenómeno de los llamados de aprendizaje (el usuario aprende el funcionamiento de una técnica, por lo que la siguiente vez le llevará menos tiempo realizar la misma tarea) sucede con las técnicas Ray-Casting y Ray-Casting with reeling, dado que la mayoría de usuarios realizaban en primer lugar las pruebas con la primera y luego con la segunda (y su funcionamiento es idéntico para tareas de selección). Puede observarse que la técnica de Ray-Casting with Reeling obtiene tiempos menores que la Ray-Casting.

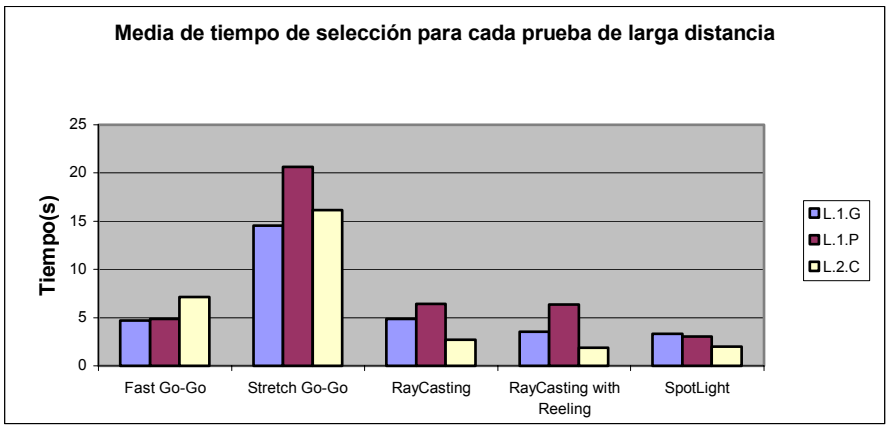


Fig. 5. Media del tiempo de selección para cada una de las pruebas a larga distancia para cada técnica.

Para larga distancia, como cabría esperar, las técnicas de puntero virtual obtienen mejores resultados (en general) que las de extensión de brazos, pero se observa que en la prueba L.1.P los tiempos de selección se elevan por encima de Fast Go-Go, debido a que el objetivo a seleccionar era relativamente pequeño, lo que hacía difícil apuntarle con el láser (lo que no aparece con la técnica SpotLight al disponer de un volumen de selección).

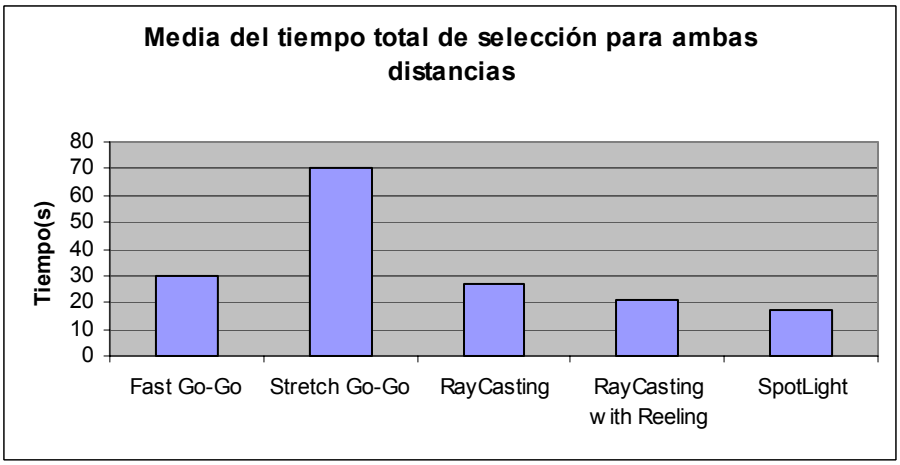


Fig. 6. Media del tiempo total de selección a ambas distancias para cada técnica.

Los resultados para las técnicas a ambas distancias son similares a obtenidos con anterioridad, Stretch Go-Go es, por mucho, la que peores tiempos obtiene, dada su mayor dificultad de manejo. Fast Go-Go se sitúa cerca de las técnicas de puntero

virtual, pero SpotLight se consolida como la técnica que menor tiempo de selección obtiene, debido de nuevo al volumen cónico.

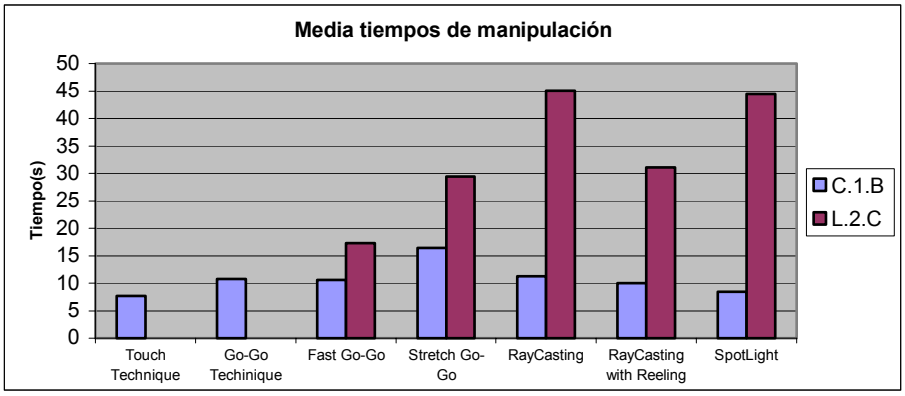


Fig. 7. Media del tiempo de manipulación para cada técnica.

Para corta distancia se observan unos tiempos similares para todas las técnicas, debido sobre todo a que la manipulación a llevar a acabo era sencilla, sólo había que cambiar la posición del objeto, pero no su orientación o la distancia al usuario, por lo que las técnicas de puntero virtual no se veían claramente perjudicadas. A pesar de todo, la técnica que mejor media obtiene es Touch Technique, algo comprensible, dado la similitud de esta técnica con la interacción en el mundo real. Nuevamente, la que peor media obtiene vuelve a ser Stretch Go-Go.

Hay que comentar que las técnicas de puntero virtual (salvo Ray-Casting with Reeling) se comportaban muy mal por naturaleza en la prueba de manipulación a larga distancia, dado que ahora sí había que cambiar la orientación del objeto y su profundidad con respecto al usuario. Por lo que el llegar a completar la prueba con Ray-Casting y SpotLight era prácticamente imposible, ya que requería mucha más paciencia y esfuerzo por parte del usuario, que tenía que acercar el objeto hacia él en varios pasos.

Aquí se observa la facilidad mostrada por las técnicas de extensión de brazos para la manipulación de objetos, la que mejor media obtiene es Fast Go-Go, seguida de Stretch Go-Go. Tras ésta, y con una media similar, se encuentra Ray-Casting with Reeling, que requería que el usuario acercara el cuadro hacia él, y luego que lo rotara, si bien la técnica daba facilidades para la primera acción, en la segunda no mostraba ninguna con respecto a las demás técnicas de puntero virtual.

Hay que comentar aquí que la varianza y la desviación típica (que nos indican lo alejados que están los valores de la media) en la medida del tiempo de manipulación son muy altas, debido sobre todo a que había participantes en el experimento que pretendían colocar los objetos con más exactitud que otros.

Tabla 2. Media, desviación típica y varianza en los tiempos de manipulación

Técnica	Prueba	Media	D. típica	Varianza
Touch Technique	C.1.B	7,63	3,76	14,16
Go-Go Techinique	C.1.B	10,82	6,61	43,70
Fast Go-Go	C.1.B	10,56	4,45	19,83
	L.2.C	17,29	5,31	28,22
Stretch Go-Go	C.1.B	16,45	10,53	110,85
	L.2.C	29,44	10,62	112,68
RayCasting	C.1.B	11,25	4,80	23,00
	L.2.C	45,02	9,42	88,81
RayCasting with Reeling	C.1.B	9,99	4,62	21,34
	L.2.C	31,10	14,93	222,98
SpotLight	C.1.B	8,49	3,63	13,21
	L.2.C	44,43	7,85	61,56

Observando los resultados de la tabla 4.4 hay que tener en cuenta las unidades en las que se nos muestran, la media es en segundos, la desviación típica también lo es, y la varianza es en segundos al cuadrado, por las peculiaridades del modo en el que se calcula este valor estadístico, y al tratarse del cuadrado de la desviación típica.

4 Conclusiones

Tras la implementación de las mismas y su evaluación a través del experimento diseñado a tal efecto, se está en posición de valorar el trabajo realizado en su conjunto, y sacar las conclusiones más importantes a las que se ha llegado, las cuales son las siguientes:

- No existe una técnica que sea mejor que las demás en todos los sentidos, ya que para ciertas tareas unas son mejores que otras. Por ejemplo, técnicas pertenecientes a la metáfora de mano virtual (como puede ser Fast Go-Go) han dado muy buenos resultados a la hora de manipular objetos. Pero las técnicas pertenecientes a la metáfora de rayo virtual han dado mejores resultados a la hora de seleccionar objetos, algo que se hacía todavía más evidente con la distancia.
- Para arrojar más luz sobre este campo es necesario un estudio mucho más ambicioso, dado que 36 pruebas realizadas por 10 participantes dejan un margen muy corto de resultados a los que aplicar técnicas estadísticas más sofisticadas, como pueden ser distribuciones de probabilidad.
- Las de manipulación deben considerarse como tareas distintas, ya que, de entre las técnicas analizadas, no había ninguna que se comportase correctamente para ambas.

Agradecimientos

Agradecemos la inestimable ayuda prestada para este proyecto por D. Federico Botella Beviá y D. Enrique Lazcorreta Puigmartí, profesores de la Universidad Miguel Hernández de Elche. Este artículo ha sido financiado por el proyecto de la Junta de Comunidades de Castilla-La Mancha PBC-03-003 de "Metodologías de desarrollo de interfaces de usuario dinámicas".

Referencias

1. [Bowman et al, 1997] *An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments*. D. Bowman y L. Hodges. Proceedings of ACM Symposium on Interactive 3D Graphics, 35-38. También disponible en http://people.cs.vt.edu/~bowman/papers/grab.pdf. 1997.
2. [Bowman, 1999] *Interaction Techniques For Common Tasks In Immervise Virtual Environments Design, Evaluation, and Application*. Tesis doctoral de D. Bowman, Georgia Institute of Technology, 67-87. También puede encontrarse en: http://people.cs.vt.edu/~bowman/. 1999.
3. [Flasar, 2000] *3D Interaction in Virtual Environment*. Jan Flasar. También disponible en http://www.cg.tuwien.ac.at/studentwork/CESCG/CESCG-2000/JFlasar/paper.pdf. 2000.
4. [Forsberg et al, 1996] *Aperture based selection for immersive virtual environment*. A. Forsberg, K. Herndon y R. Zeleznik. Proceedings of UIST'96, pp. 95-96. Disponible también en: http://portal.acm.org/citation.cfm?id=237105&dl=ACM&coll=portal. 1996.
5. [Mine, 1995] *Virtual Environment Interaction Techniques*. M. Mine. UNC Chapel Hill Computer Science Technical Report TR95-018. También disponible en http://www.cs.unc.edu/~mine/mine_publications.html. 1995.
6. [Poupyrev et al, 1996] *The Go-Go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR*. I. Poupyrev, M. Billinghurst, S. Weghorst y T. Ichikaw. Proceedings of ACM Symposium on User Interface Software and Technology, 79-80.Disponible también en la siguiente URL: http://www.hitl.washington.edu/people/poup/research/papers/uist96.pdf. 1996.
7. [Poupyrev et al, 1997] *A Framework and Testbed for Studying Manipulation Techniques for Immersive VR*. I. Poupyrev, M. Billinghurst, S. Weghorst y T. Ichikaw. Proceedings of ACM Symposium on Virtual Reality Software and Technology, 21-28. Disponible también en la siguiente URL: http://www.hitl.washington.edu/people/poup/research/papers/vrst97.pdf. 1997.
8. [Poupyrev et al, 1998] *A study of techniques for selecting and positioning objects in immersive VEs: effects of distance, size, and visual feedback*. Ivan Poupyrev, Suzanne Weghorst, Mark Billinghurst y Tadao Ichikawa. Presented at ACM CHI '98. También disponible en http://www.hitl.washington.edu/publications/r-97-45/r-97-45.pdf. 1998.

Navegación mediante Web Semántica: Una Nueva Arquitectura

Oscar Martínez[1], Federico Botella[1] y Antonio Fernández-Caballero[2]

[1]Centro de Investigación Operativa
Universidad Miguel Hernández de Elche, 03202 Elche
{oscar.martinez, federico}@umh.es
[2]Laboratory of User Interaction and Software Engineering
Computer Science Research Institute
University of Castilla-La Mancha, Albacete (Spain)
caballer@info-ab.uclm.es

Resumen. La Web Semántica se ha convertido en un área de investigación importante donde confluyen numerosas tecnologías Web actuales. Dichas tecnologías proponen la introducción de descripciones concretas sobre el significado de los recursos para permitir que las propias máquinas tengan un nivel de comprensión de la Web con el suficiente detalle como para recuperar información relevante para el usuario que está ejecutando una búsqueda en la Web. En este artículo proponemos una nueva arquitectura para la navegación y la visualización de información relevante dentro de un dominio experimental concreto, orientado al comercio electrónico de prendas de moda. Por último, se describe cómo la aplicación de determinadas técnicas de la Web Semántica nos permite la mejorar el diseño de las distintas interfaces de usuario implementandas utilizando además los conceptos de adaptabilidad y de usabilidad.

1 Introducción

Actualmente, la Web es un espacio preparado para el intercambio de información diseñado para el consumo humano. Es más, las páginas Web son creadas por personas para ser entendidas por personas. No existe un formato estándar para mostrar la información, por la cual, los desarrolladores de páginas Web crean sus páginas dependiendo de los usuarios potenciales que van a visitarlas posteriormente. En este contexto, la Web se puede considerar como un gigantesco texto distribuido en una red de baja calidad, con contenido pobremente escrito, no focalizado, sin una buena organización y consultado por los usuarios más inexpertos. Puesto que comprender cabalmente el significado de un texto en forma automática es imposible en la práctica, los sistemas de recuperación de información buscan una aproximación a lo que el usuario está buscando; de esta manera, seleccionar un buen sistema de recuperación de información consistirá en aquél que permita hacer las preguntas adecuadas y como resultado entender las respuestas [1].

Con toda esta evolución de la Web, a finales de los 90 surge la visión de lo que se ha denominado *Web semántica* [2]. Se trata de una corriente, promovida por el propio inventor de la Web y presidente del consorcio *World Wide Web Consortium* (W3C), cuyo último fin es lograr que las máquinas puedan entender, y por tanto utilizar, lo que la Web contiene. Esta nueva Web estaría poblada por agentes software capaces de navegar y realizar operaciones por el usuario para ahorrar trabajo y optimizar los resultados. Para conseguir esta meta, la Web semántica propone describir los recursos de la Web con representaciones procesables (es decir, entendibles) no sólo por personas (usuarios), sino por programas que puedan asistir, representar, o reemplazar a las personas en tareas rutinarias o inabarcables para un humano. Las tecnologías de la Web semántica buscan desarrollar una Web más cohesionada, donde sea aún más fácil localizar, compartir e integrar información y servicios, para sacar un partido todavía mayor de los recursos disponibles en la Web.

En la sección 2 se presentan diferentes técnicas de navegación por la Web Semántica, introduciendo el uso de ontologías. La sección 3 presenta la arquitectura propuesta para la gestión de contenidos relacionados con la compra de prendas de moda, donde se aplicaremos ciertas técnicas basadas en la Web Semantica. En esta plataforma se pretende proporcionar la siguiente funcionalidad: adaptabilidad, búsqueda, visualización y navegación basadas en la web semántica. Por último, en la sección 4 se presentan las conclusiones y el trabajo futuro.

2 Las ontologías como soporte a la navegación en Web Semántica

Aunque es sumamente difícil medir el tamaño de la Web, se estima que actualmente unos 10^9 usuarios utilizan la Web, y que ésta contiene del orden de $4 \cdot 10^9$ documentos [3]. Estas cifras incluyen sólo los documentos estáticos accesibles en la Web. Se ha calculado que la parte constituida por las bases de datos cuyos contenidos, no directamente accesibles, se hacen visibles mediante páginas generadas dinámicamente, puede contener un tamaño de información varios cientos de veces mayor, y de mucha mejor calidad, que la parte estática comentada en primer lugar, y crece a un ritmo aún mayor que ésta [4]. Se estima que el tamaño de la parte dinámica ha superado ya al volumen total de información impresa existente en todo el planeta.

En este contexto un problema clave es alcanzar un entendimiento entre las partes que han de intervenir en la construcción y explotación de la Web tales como usuarios, desarrolladores y programas de muy diverso perfil. La Web Semántica utiliza la noción de *ontología* [5] como vehículo para cumplir este objetivo, la cual se puede definir como una jerarquía de conceptos con atributos y relaciones, que define una terminología consensuada para definir redes semánticas de unidades de información interrelacionadas. Una ontología proporciona un vocabulario de clases y relaciones para describir un dominio, poniendo el acento en la compartición del conocimiento y el consenso en la representación de éste. Hay diversos lenguajes formales para definir ontologías entre los que destacan de manera importante RDF [6], DAML+OIL [7] y OWL [8]. Además, escribir en lenguajes como RDF y OWL es relativamente complicado y propenso a errores. Para evitar esto último, existen programas gráficos que visualizan y construyen ontologías de forma intuitiva como *Protégé* [9].

Para entender mejor todo lo anterior, definimos como ejemplo una ontología sobre moda la cual podría incluir clases como Modisto, Vestido, Estilo o Tienda, y relaciones como *diseñador* de un vestido, modistos *pertenecientes* a un estilo de moda o vestidos *localizados* en una tienda. La idea es que, una vez establecida una ontología sobre prendas de moda, la Web Semántica estará formada por una red de nodos tipificados e interconectados mediante clases y relaciones definidas por una ontología compartida por todos sus autores. A continuación, una tienda de moda podría organizar sus contenidos definiendo instancias de diseñadores, trajes, etc., interrelacionándolas y publicándolas en una Web Semántica. En la figura 1 se muestra la representación de cada nodo (recurso) de un tipo (modisto, escuela, vestido, etc.), y las relaciones, representadas por los arcos, explícitamente diferenciadas (modisto-escuela, modisto-vestido, etc.).

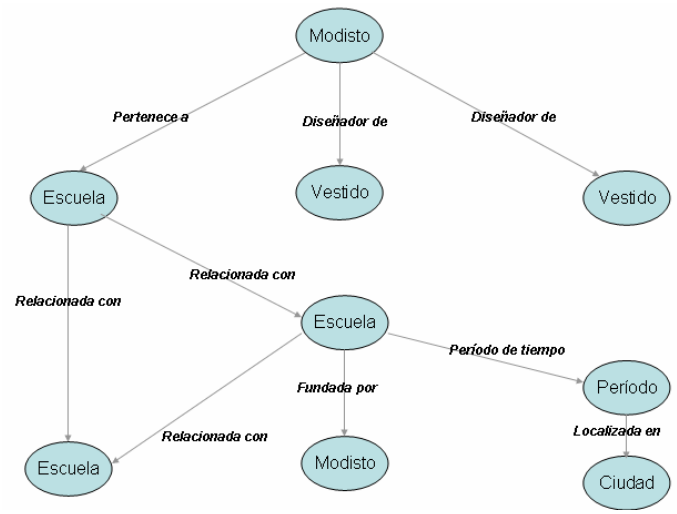


Fig. 1. Representación de la ontología en la Web Semántica.

La adopción de ontologías comunes es clave para que todos los agentes que participen de la Web Semántica, contribuyendo o consumiendo recursos, puedan trabajar de forma autónoma con la garantía de que las piezas encajen. En nuestro ejemplo, varias escuelas de modistos podrían colaborar para dar lugar a una gran meta-escuela que integre los contenidos de todas ellas.

Un programa que navegue por un portal como éste puede reconocer las distintas unidades de información, obtener datos específicos o razonar sobre relaciones complejas. A partir de aquí sí que se puede distinguir entre un vestido *diseñado por* un modisto y un vestido *de* un modisto.

Por último, generalmente la Web no sólo proporciona acceso a contenidos sino que también ofrece interacción y servicios (comprar un vestido, reservar una prenda de pretemporada, realizar una transferencia bancaria para pagar un volumen considerable de prendas, etc.). Los servicios Web Semánticos son una línea importante de la Web Semántica, que propone describir no sólo información sino

definir ontologías de funcionalidad y procedimientos para describir servicios Web: sus entradas y salidas, las condiciones necesarias para que se puedan ejecutar, los efectos que producen, o los pasos a seguir cuando se trata de un servicio compuesto. Estas descripciones procesables por máquinas permitirían automatizar el descubrimiento, la composición, y la ejecución de servicios, así como la comunicación entre unos y otros.

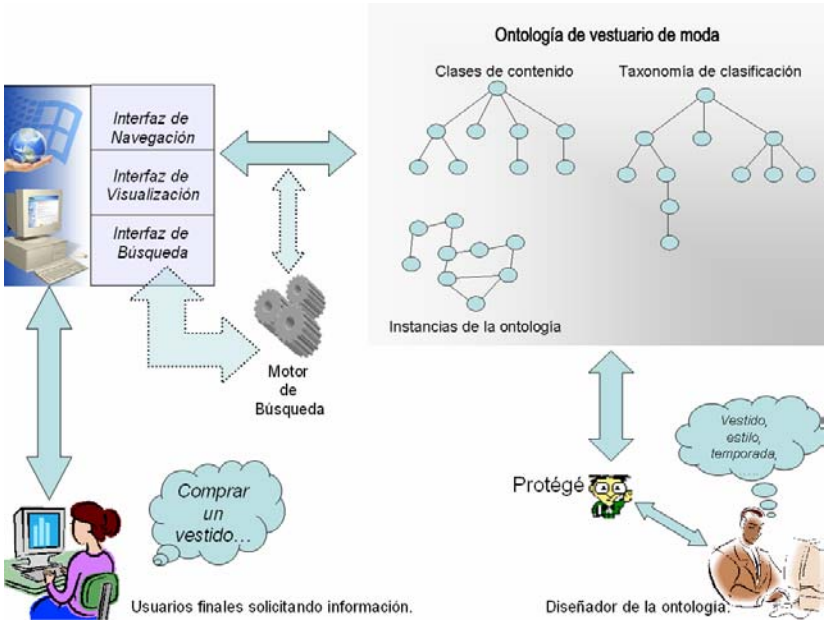


Fig. 2. Arquitectura propuesta.

3 Arquitectura propuesta

Las empresas dedicadas a la venta de productos cuya información pertenece a este dominio, se podrían beneficiar de las ventajas aportadas por la Web Semántica. Estas técnicas permiten la estructuración del dominio así como la búsqueda de información de una manera mucho mas refinada que las técnicas tradicionales. A continuación, presentamos una arquitectura propuesta para la navegación en webs semánticas, en concreto se ha implementado un portal Web denominado "*e-fashion"* cuya temática está directamente relacionada con compra-venta de prendas de moda. En esta arquitectura, expuesta en la figura 2, los usuarios finales pueden interactuar con el sistema a través de un *interfaz de navegación* similar a cualquier navegador de Internet convencional, lo que permite realizar consultas al sistema seleccionando un contenido de la ontología (vestido, camisa, pantalón, etc.). Como respuesta, el sistema muestra los atributos del contenido seleccionado a través del *interfaz de visualización* para que el usuario rellene los datos que le resulten más relevantes. Adicionalmente,

también puede seleccionar la categoría en la que está clasificado el contenido a través del *interfaz de búsqueda*. Una vez formulada la consulta se envía al sistema y éste devuelve los resultados adecuados según los valores devueltos por el *motor de búsqueda*. El resultado devuelto, tal como puede apreciarse en la figura 3, puede que contenga instancias de diferentes tipos, por lo que cada vez que se seleccione una instancia, es el modulo de visualización el encargado de mostrarlas adecuadamente.

Fig. 3. Resultado de una consulta.

Con respecto al diseño del interfaz Web, para representar la información de la propia ontología, la arquitectura propuesta aplica el concepto de adaptabilidad con el fin de ofrecer una navegación semántica "inteligente" que adapta el interfaz según el perfil de usuario que se encuentra utilizando la aplicación. Esta característica se consigue mediante la inclusión de técnicas basadas en distancias conceptuales lo que permite mostrar visualmente conceptos próximos [10].

Por último, para la creación y el modelado de la ontología en el dominio de las prendas de moda se ha utilizado la herramienta *Portégé*. Con respecto al lenguaje de consultas hemos escogido RDFS y RDF, siendo éste último uno de los más asentados y ampliamente soportado por múltiples herramientas de manejo de ontologías. De esta manera, el *diseñador de la ontología* usa RDFS para la definición de los conceptos (clases), atributos, relaciones de la ontología y RDF para las instancias. En la figura 4, se muestra un fragmento de la definición de la ontología utilizada en esta arquitectura.

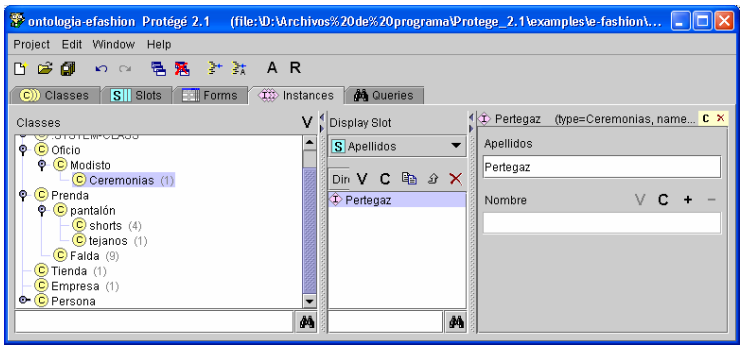


Fig. 4. Definición de la ontología con Protégé.

4 Conclusiones y Trabajo Futuro

En este trabajo se ha presentado una nueva arquitectura que permite mejorar la navegación en un portal web semántico centrado en un dominio concreto orientado a las prendas de moda. La arquitectura propuesta apuesta por el uso de RDF/RDFS, de manera que los resultados obtenidos hasta el momento han mostrado una buena capacidad de conceptualización. Concretamente, se ha aportado una nueva técnica por la que la precisión de búsqueda aporta mejoras considerables con respecto a las búsquedas tradicionales.

Como trabajo futuro se tiene pendiente el uso de otros lenguajes para la definición de ontologías como DAML-OIL o OWL puesto que tienden a ser lenguajes estándares para la Web Semántica. Además, se ha iniciado la implementación de servicios Web Semánticos con el fin de desarrollar agentes personales que permitan refinar nuestro modelo de navegación en Web Semántica.

Agradecimientos

Este trabajo está financiado en parte por el proyecto de la Junta de Comunidades de Castilla-La Mancha PBC-03-003.

Referencias

1. Baeza, R., Ribeiro, B., Modern Information Retrieval, Addison-Wesley, 1999.
2. Berners-Lee, T., Hendler, J., Lassila, O., The Semantic Web. Scientific American May 2001.
3. Bergman, M. K., The Deep Web: Surfacing Hidden Value. The Journal of Electronic Publishing. Volume 7, Issue 1, August, 2001.
4. O'Neill, E. T., Lavoie, B. F., Bennett, R., Trends in the Evolution of the Public Web. D-Lib Magazine, Volume 9 Number 4, April 2003.

5. Gruber, T. R., A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, 5(2), pp. 199-220, 1993.
6. Lassila, O., Swick, R. R., Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation 22 February 1999. Available at http://www.w3.org/TR/REC-rdf-syntax.
7. Connolly, D., Van Harmelen, F., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F. and Stein, L. A., DAML+OIL Reference Description. W3C Note 18 December 2001. Available at http://www.w3.org/TR/daml+oil-reference.
8. Dean, M., Connolly, D., Van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F. and Stein, L. A. , OWL Web Ontology Language 1.0 Reference W3C Working Draft 29 July 2002. Available at http://www.w3.org/TR/owl-ref.
9. Protégé project. Available at http://protege.semanticweb.org
10. Peñarrubia, A., Fernández-Caballero, A., González, P., Botella, F., Grau, A. and Martínez, O., Ontology-based interface adaptivity in web-based learning systems, Accepted, 4th IEEE International Conference on Advanced Learning Technologies, ICALT'04, September 2004.

Uso de Técnicas de Data Mining para la Clasificación de Sesiones de Navegación, a partir de los Ficheros de Registro de un Servidor

Alejandro Rabasa[1], Enrique Lazcorreta[1], Federico Botella[1] y Antonio Fernández-Caballero[2]

[1]Centro de Investigación Operativa
Universidad Miguel Hernández de Elche, 03202 Elche
{a.rabasa, enrique, federico}@umh.es
[2]Laboratory of User Interaction and Software Engineering
Computer Science Research Institute
University of Castilla-La Mancha, Albacete (Spain)
caballer@info-ab.uclm.es

Resumen. Los ficheros de registro de entradas de cualquier servidor web contienen información muy valiosa sobre los perfiles de navegación de los usuarios. Para encontrar patrones de comportamiento en la navegación a partir de estos ficheros es necesaria una minuciosa etapa de preprocesamiento en la que importemos esos datos a una tabla libre de redundancias y de entradas no-relevantes. En esta fase adecuaremos, también, los tipos de datos a las necesidades de los algoritmos de Minería y segmentaremos nuestro espacio de trabajo, encontrando valores discriminatorios críticos que ayudarán a crear grupos de entradas, atendiendo a diferentes criterios (status, fecha…). En este trabajo se muestran los resultados de una serie de pruebas basadas en modelos, en la etapa de preprocesamiento y se señalan las líneas futuras para el desarrollo de algoritmos que ayuden a la clasificación de sesiones y/o usuarios.

1 Introducción

Las sesiones de navegación a través de las páginas web de un determinado servidor generan una serie de entradas en los correspondientes ficheros de registro, los cuales pueden ser parametrizados y analizados a partir de diferentes herramientas existentes en el mercado (Analog[1], Logparser[2] o Webalizer[3], entre otras). Sin embargo, la clasificación sistemática de dichas sesiones o usuarios, en función de sus perfiles de navegación, no se puede llevar a cabo de manera inmediata con el mero uso de estas herramientas.

En este trabajo nos fijamos como objetivo realizar un primer análisis sobre las posibilidades del Data Mining para la identificación de patrones de comportamiento

[1] http://www.analog.cx/
[2] http://www.microsoft.com/technet/community/scriptcenter/logs/logparser/default.mspx
[3] http://www.webalizer.org/

en las sesiones de navegación, y para ello evaluaremos el uso del módulo Analysis Services de Microsoft SQL Sever 2000, con el que fragmentaremos el espacio de datos mediante la generación de modelos de Árboles de Decisión.

La entrada a nuestro sistema de Minería será un fichero de registro preprocesado previamente. Esta etapa de preprocesamiento se llevará a cabo mediante las herramientas de parametrización y análisis de logs, y su finalidad será proporcionar un fichero de registros libre de errores y de datos no-relevantes para el problema de la clasificación de sesiones.

Los modelos de Árboles de Decisión generados se "cruzarán" contra el fichero de registro preprocesado, de manera que sobre la estructura de árbol proporcionada por el sistema seamos capaces de dividir el espacio de datos de entrada, en función de los valores críticos encontrados por los modelos.

En general, podemos distinguir dos tipos de sesiones:

- *Sesiones identificadas*: El usuario se identifica al acceder al servidor. Se podrá hacer un seguimiento al usuario a través de diferentes sesiones de navegación.
- *Sesiones no-identificadas o anónimas*: El usuario no se identifica tras acceder al servidor. En estas circunstancias tan sólo se podrá, en principio, clasificar las sesiones de navegación (y no a los usuarios).

Este primer estudio se centra en el caso de sesiones anónimas, de las que intentaremos extraer información sobre los *status* más devueltos por el servidor, las franjas horarias y fechas con más accesos. En posteriores trabajos nos centraremos en el uso de las sesiones identificadas para buscar un seguimiento más próximo de las preferencias del usuario.

Las pruebas realizadas se han llevado a cabo sobre los registros del servidor del Dpto. de Estadística y Matemática Aplicada de la Universidad Miguel Hernández durante el primer trimestre de 2004.

2 Generación de Modelos de DM para la Segmentación de los Datos

2.1 Modelos proporcionados por SQL Server 2000

El módulo Analysis Services de Microsoft SQL Server 2000, [4] proporciona dos tipos de modelos para la Minería de Datos:

- Clustering
- Árboles de Decisión

El modelo de Clustering, basado en el algoritmo *k-mean*, proporciona 10 grupos de registros que cumplen ciertos criterios. Dichos grupos no son excluyentes, es decir, hay registros que pertenecen a más de un grupo. Los criterios por los que el algoritmo decide su pertenencia o no al grupo, se basan en el grado de proximidad a los valores "centro" calculados para algunos campos tras varias iteraciones. En este proceso no

se le permite al usuario ponderar la relevancia de cada campo, ni el número de clusters a formarse. El usuario sólo puede decidir cuáles son los campos de entrada al modelo, en concreto, aquéllos en base a los cuáles se generan los clusters. Los modelos de Clustering se emplean sólo para problemas de clasificación.

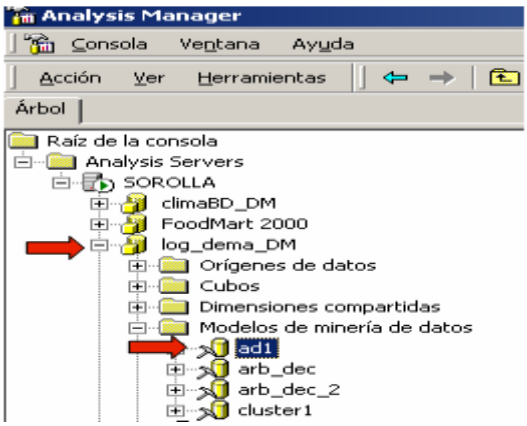


Fig. 1. Pantalla Analysis Manager, con los primeros modelos de prueba

Por otro lado, el modelo Árboles de Decisión, bajo el nombre de *Microsoft Decisión Trees*, y basado en los algoritmos *CART, CHAID y C4.5* [5], fragmenta el conjunto total de datos en una estructura ramificada donde las hojas de la misma sí que suponen grupos excluyentes, encontrando valores discriminatorios entre los campos denominados "de entrada" y calculando la frecuencia de aparición de valores discretos (o discretizados) entre los denominados "campos de predicción". Cada nodo de la estructura Árbol de Decisión contiene las condiciones críticas, un conteo de cuántos registros las cumplen y un porcentaje de presencia para cada uno de los valores posibles del campo denominado "de predicción". Los modelos de Árboles de Decisión se pueden emplear tanto en problemas de clasificación como en problemas de predicción.

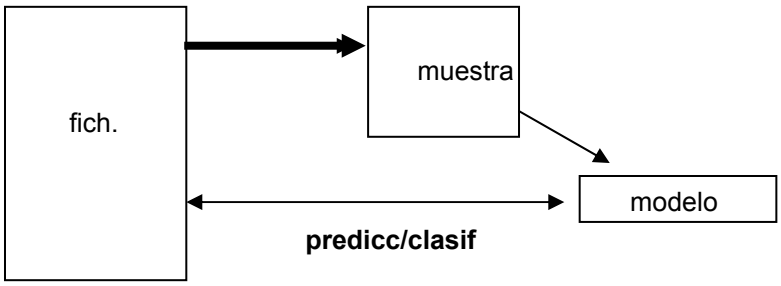


Fig. 2. Esquema del proceso completo de minería en Analysis Services.

Nos centraremos en los Árboles de Decisión, ya que trataremos de identificar los valores más significativos que definen nuestros datos (Minería descriptiva). No olvidemos que la finalidad última será la clasificación de las sesiones, atendiendo a diferentes criterios que cumplan los "campos de entrada", para lo cual es muy conveniente tener identificados los valores más frecuentes entre ellos.

En la figura 2 se muestra el esquema del proceso de Minería, empleando Analysis Services de Microsoft SQL Server 2000. A partir del fichero de logs se extrae una muestra sobre la cual generamos los modelos Árbol de Decisión. La explotación de tales modelos (a través de algoritmos propios o de consultas denominadas "Prediction Queries") consiste en "cruzar" los datos del modelo con el fichero de logs que entró al sistema.

```
SELECT FLATTENED
  [T1].[Datetime ip], [T1].[Date time], [T1].[C ip], [T1].[Sc status],
[AD1_status_ip_datetime].[Sc status]
FROM
  [AD1_status_ip_datetime]
  PREDICTION JOIN
  OPENROWSET
  (
    'SQLOLEDB.1',
    'Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security
Info=False;Initial Catalog=log_dema;Data Source=SOROLLA',
    'SELECT "datetime-ip" AS "Datetime ip", "date-time" AS "Date time", "c-ip"
AS "C ip", "sc-status" AS "Sc status" FROM "lg04" ORDER BY "datetime-ip"'
  )
  AS [T1]
  ON
    [AD1_status_ip_datetime].[Datetime ip] = [T1].[Datetime ip] AND
    [AD1_status_ip_datetime].[Date time] = [T1].[Date time] AND
    [AD1_status_ip_datetime].[C ip] = [T1].[C ip] AND
    [AD1_status_ip_datetime].[Sc status] = [T1].[Sc status]
```

Fig. 3. Ejemplo de código de Prediction Query, donde "AD1_status_ip_datetime" es el modelo generado y "T1" es la tabla con todos los logs.

2.2 Generación de Árboles de Decisión

2.2.1 Elección de la muestra

Debido a que la explotación de los modelos se hace cruzando el conjunto total de los datos con la estructura resultante de la muestra, ésta no puede ser el total de los registros del fichero de logs. La elección de dicha muestra se presenta como un proceso en el que debemos ser cuidadosos de no eliminar registros que sean especialmente significativos, ni tampoco debemos eliminar registros consecutivos,

puesto que podríamos estar eliminando fragmentos de sesiones, o con una misma franja horaria de navegación.

Una primera aproximación es quedarse con 3/4 (aproximadamente) de los datos iniciales, tomando una selección aleatoria de sesiones a eliminar.

2.2.2.- Procedimiento de generación de Árboles de Decisión

Además de elegir el origen de datos (muestra), el proceso de generación de Árboles de Decisión deberá ser parametrizado indicando qué campos de los disponibles van a ser campos de entrada y qué otros campos van a ser campos de predicción.

En este contexto consideramos el término "predicción" como la frecuencia (de ocurrencia) con que dicho campo toma cada uno de sus posibles valores; por lo cual, los campos de predicción deberán ser del tipo *"discrete"* (discreto) o *"discretized"* (discretizado).

Una vez parametrizado el modelo debemos procesarlo. Durante esta fase, el algoritmo *Microsoft Decisión Trees* encuentra los valores discriminatorios de los campos de entrada sobre la muestra proporcionada, y realiza un conteo de las veces que el campo de predicción toma cada uno de sus posibles valores dentro de cada grupo.

Cualquier variación en los campos de entrada o de predicción (por incorporación de campos, por supresión de campos o por modificación de sus tipos) obliga al re-procesado completo del modelo. De hecho, se tratará de un árbol de decisión diferente, y como tal deberá ser guardado y analizado.

3 Clasificación de Sesiones de Navegación

3.1 Preprocesamiento

Para conocer los datos almacenados por el servidor hemos comenzado usando Analog, que nos ofrece una rápida descripción estadística de los ficheros de registro:

- Informe mensual: actividad (peticiones al servidor) de cada mes.
- Resumen diario: resumen de la actividad para cada día de la semana.
- Resumen horario: resumen de la actividad para cada hora del día.
- Informe de organización: resumen de las raíces de las IP's de los clientes
- Informe de sistemas operativos.
- Informe de códigos de estado.
- Informe de tipos de archivo.

El análisis de esta información nos permite ver las características del servidor a partir de los ficheros de registro.

Después de revisar el informe proporcionado por *Analog* decidimos filtrar los ficheros de registros con *LogParser*, que permite realizar consultas SQL y obtener nuevos ficheros como resultado de dichas consultas con datos ya filtrados. Decidimos

trabajar con un fichero de registro que sólo contenga peticiones de páginas ASP, HTM y HTML, puesto que en este servidor son las únicas realmente solicitadas por los usuarios.

Usaremos Analog de nuevo para observar la actividad del servidor una vez filtrados los ficheros de registro. Esto nos puede llevar a la decisión de filtrar de nuevo los ficheros de registro en función de los resultados obtenidos.

En esta fase de preprocesamiento ejecutaremos iterativamente las fases de:

1) Análisis
2) Filtrado

hasta que el fichero de registros obtenido contenga la información relevante para nuestro estudio. Es decir, que cada línea del registro se identifique con una página web (y no con cada uno de sus posibles componentes), y que todos los campos sean significativos (que el tipo de datos sea el adecuado). Este proceso se describe con más detalle en el punto 2.2.

3.2 Importación de datos desde SQL Server 2000

Con Logparser se genera el fichero de registros de nombre *pr3.log*, el cual es importado a nuestra base de datos *log_dema* en SQL Server 2000, sobre la tabla *logpwebX*.

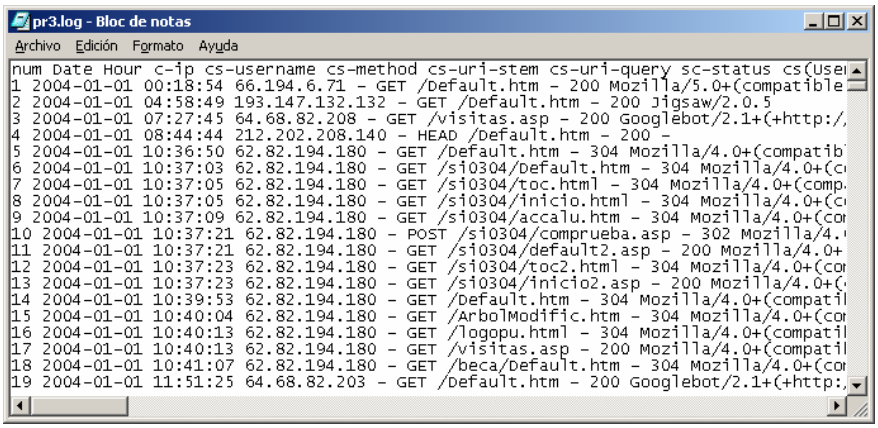


Fig. 4. Fichero texto plano pr3.log generado por LogParser

En la tabla importada cambiamos sus tipos de datos por defecto por el tipo más adecuado en cada caso y generamos claves e índices. Además añadimos varios campos que serán necesarios en el procesamiento posterior, tal como se describe a continuación:

Los campos *Date-Hour, c-ip, cs-username, cs-method, cs-uri-stem, cs-uri-query, sc-status, cs(User-Agent)* se importan automáticamente con un tipo de datos genérico

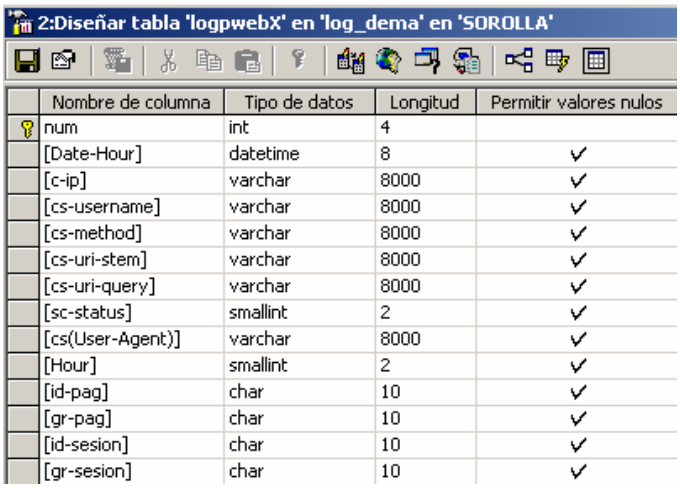
denominado *varchar*. Dicho tipo es cambiado por otros más adecuados para su explotación (ver figura 5).

Se ha generado con LogParser el campo *num* (autonumérico) que será la clave principal de la tabla.

Para atender las necesidades de identificar exclusivamente la hora (como valor discreto) se añade *Hour* como campo numérico resultante de extraer la subcadena de la hora de *Date-Hour*.

Se añaden también algunos campos que serán imprescindibles para el descubrimiento de patrones:

- *id-pag*: Identificador de la página.
- *gr-pag*: Grupo de la página. (Tras un proceso de generalización) [2], [3]
- *id-sesion*: Identificador de sesión. [3]
- *gr-sesion*: Grupo de sesión



Nombre de columna	Tipo de datos	Longitud	Permitir valores nulos
num	int	4	
[Date-Hour]	datetime	8	✓
[c-ip]	varchar	8000	✓
[cs-username]	varchar	8000	✓
[cs-method]	varchar	8000	✓
[cs-uri-stem]	varchar	8000	✓
[cs-uri-query]	varchar	8000	✓
[sc-status]	smallint	2	✓
[cs(User-Agent)]	varchar	8000	✓
[Hour]	smallint	2	✓
[id-pag]	char	10	✓
[gr-pag]	char	10	✓
[id-sesion]	char	10	✓
[gr-sesion]	char	10	✓

Fig. 5. Diseño final de la tabla *logpwebX*

Después de realizar estos cambios en el diseño, la tabla queda con los siguientes campos:

- *num*: campo clave numérico generado automáticamente por Logparser.
- *Date-Hour*: campo resultante de concatenar *Date* y *Hour* del fichero proporcionado
- *c-ip*: dirección IP del cliente.
- *cs-username*: nombre del usuario, si se valida en el servidor. Casi siempre en blanco.
- *cs-method*: Método empleado por el cliente.
- *cs-uri-stem*: Recurso solicitado por el cliente.
- *cs-uri-query*: consulta del cliente.
- *sc-status*: estado devuelto por el servidor al cliente tras su solicitud.
- *cs(User-Agent)*: agente empleado por el cliente.

- *Hour*: campo numérico resultante de extraer de *Date-Hour*, la subcadena de la hora.
- *id-pag*: Identificador de la página.
- *gr-pag*: Grupo de la página.
- *id-sesion*: Identificador de sesión.
- *gr-sesion*: Grupo de sesión.

	num	Date-Hour	c-ip	cs-user...	cs-method	cs-uri-stem	cs-
1	1	2004-01-01 00:18:54.000	66.194.6.71	-	GET	/Default.htm	-
2	2	2004-01-01 04:58:49.000	193.147.132.132	-	GET	/Default.htm	-
3	3	2004-01-01 07:27:45.000	64.68.82.208	-	GET	/visitas.asp	-
4	4	2004-01-01 08:44:44.000	212.202.208.140	-	HEAD	/Default.htm	-
5	5	2004-01-01 10:36:50.000	62.82.194.180	-	GET	/Default.htm	-
6	6	2004-01-01 10:37:03.000	62.82.194.180	-	GET	/si0304/Default.htm	-
7	7	2004-01-01 10:37:05.000	62.82.194.180	-	GET	/si0304/toc.html	-
8	8	2004-01-01 10:37:05.000	62.82.194.180	-	GET	/si0304/inicio.html	-

Fig. 6. Extracto de la tabla *logpwebX*

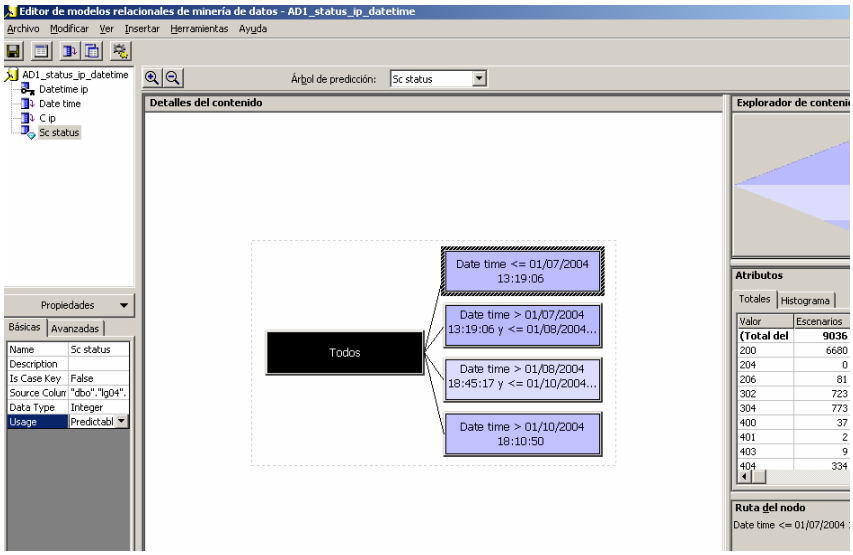


Fig. 7. Árbol de Decisión con detalles de sus componentes

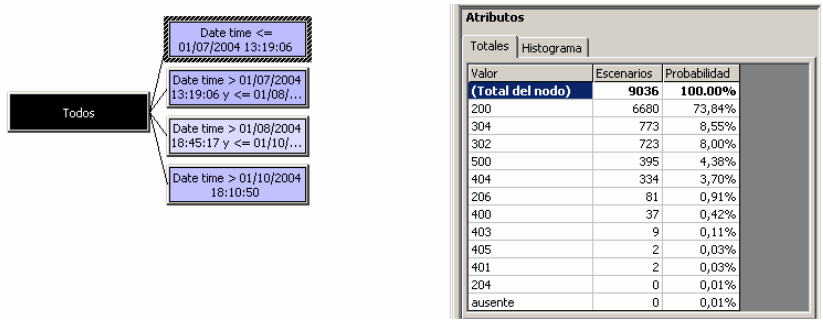
3.3 Análisis e interpretación de modelos

Como ya se ha expuesto anteriormente, los Árboles de Decisión son una herramienta dirigida principalmente a la predicción de valores ausentes. Nosotros los utilizaremos para localizar valores críticos que segmentan nuestro espacio de trabajo. Así podemos encontrar, por ejemplo, fechas en las que se da con una mayor frecuencia un cierto valor para el campo "status" o también franjas horarias con mayor número de accesos desde IPs externas.

Es decir, nos fijaremos en las mayores concentraciones de valores de ciertos campos de salida, en función de los grupos generados sobre los campos de entrada. De este modo, al igual que ocurre con el uso de las herramientas de la etapa de preprocesamiento (Logparser y Analog), conseguimos conocer mejor los datos que someteremos a procesos de Minería.

En primer lugar analizamos las *concentraciones de registros* (escenarios) en cada grupo. Se observa que:

- independientemente del grupo de fechas en que se encuadre el acceso, el status que más se devuelve por el servidor es el 200 (Respuesta OK), y en todos los grupos supera el 65% de las ocurrencias.
- Los siguientes status más devueltos son el status 304 (Redirección. No modificado) y 302 (Redirección. Cambiado temporalmente), con frecuencias próximas al 8% en ambos casos.
- También podemos utilizar esta técnica para descubrir los status que siempre se dan con menor frecuencia, como el 401 (Acceso no autorizado) con una frecuencia siempre inferior al 0,20%.



Valor	Escenarios	Probabilidad
(Total del nodo)	**9036**	**100.00%**
200	6680	73,84%
304	773	8,55%
302	723	8,00%
500	395	4,38%
404	334	3,70%
206	81	0,91%
400	37	0,42%
403	9	0,11%
405	2	0,03%
401	2	0,03%
204	0	0,01%
ausente	0	0,01%

Fig. 8. Árbol de Decisión. Conteo de status en el primer nodo.

Con la creación de modelos se percibe la importancia de utilizar valores discretos, o al menos discretizables. De no hacerlo así, y trabajando con valores continuos (como podría interpretarse inicialmente el campo Date-Hour) los árboles generados dejan de ser útiles en la medida en que tienen una profundidad y anchura excesivas (ver figura 9).

Los campos como status (*cs-status*), fecha (*Date-Hour*) o incluso la hora (*Hour*) los usaremos como campos de entrada para clasificar las sesiones de usuario, pero también necesitaremos campos como *id_sesión* (o *id_usuario*) e *id_pag* para realizar una correcta identificación de las sesiones de usuario que nos permitan agruparlas en su clasificación correspondiente. Todos estos campos deben ser de tipo discreto o discretizable.

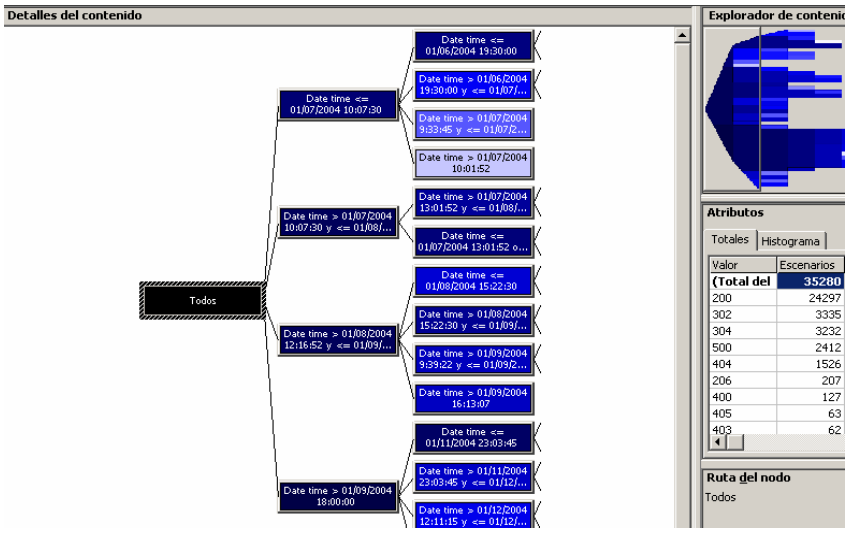


Fig. 9. Árbol de Decisión demasiado ancho y profundo, debido al uso de valores continuos.

4 Conclusiones

La etapa de preprocesamiento tiene una especial relevancia en el ámbito del Web Mining, debido a los enormes volúmenes de datos que pueden llegar a recogerse de los ficheros de logs, y a que la información almacenada en ellos no siempre está en el formato óptimo para su explotación.

Los Árboles de Decisión generados requieren campos discretos o discretizables para que los grupos de registros que nos ofrezcan sean fácilmente interpretables. Esto supone que si queremos formar grupos en función de campos continuos, éstos deberán ser discretizados previamente.

Logparser y Analog son especialmente útiles como herramientas para análisis de registros y para brindarnos una primera toma de contacto con los datos en la etapa de preprocesamiento. Dicha etapa se puede completar con el uso de SQL Server 2000 como herramienta de manipulación de datos.

Para encontrar y definir patrones de comportamiento en las sesiones de navegación, recurriremos a algoritmos de diseño propio, que nos darán una mayor flexibilidad a la hora de seleccionar los campos de entrada a los mismos y en la fase de entrenamiento de los modelos.

5 Trabajo Futuro

Para definir patrones de comportamiento y una metodología de clustering que nos permita agrupar las sesiones de navegación, comenzaremos analizando

detalladamente algoritmos de clustering conocidos como el COBWEB([10]) o el BIRCH [11]) .

Evaluaremos el uso de grafos, además de árboles de decisión, en el diseño de algoritmos de clustering y definición de patrones de comportamiento dentro de las sesiones de navegación.

Las herramientas utilizadas nos proporcionan descripciones de los ficheros de registros, pero no nos permiten analizar sesiones, por lo que estudiaremos dos posibles líneas de trabajo:

- Desarrollar una herramienta capaz de analizar los ficheros de registros y crear ficheros con información adicional sobre sesiones.
- Incluir en las páginas web de nuestro sitio, funciones específicas para completar los ficheros de registros con información susceptible de ser analizada, con la ayuda de la función AppendToLog() de ASP.NET y siguiendo las ideas sugeridas en [6] y [7].

Agradecimientos

Este trabajo está financiado en parte por el proyecto de la Junta de Comunidades de Castilla-La Mancha PBC-03-003.

Referencias

[1] "Data Mininig and Statistical Analysis Using SQL". R.P. Trueblood, J.N. Lovett. Apress, 2001.
[2] "Data Mining and uncertaining reasoning" Z. Chen. Wiley Interscience, 2001
[3] "Agrupamiento de usuarios de internet, según patrones de acceso" Yongjian Fu, et al, http://www.estadistico.com/arts.html?20020225, 2002
[4] "Data Mining with Microsoft SQL Server 2000" C. Seidman. Microsoft Press, 2001
[5] "Analysis of Data Mining Algorithms" Karuna Pande Joshi. http://userpages.umbc.edu, 1997
[6] "Tracking visitors with ASP.NET", W. Plourde, 2002, http://www.15seconds.com/issue/021119.htm
[7] "Encrypting cookie data with ASP.NET", W. Plourde, 2002, http://www.15seconds.com/issue/021210.htm
[8] "Automatic discovery of similarity relationships through Web Mining" D. Roussinov ; J. L. Zhao, Elsevier, 2003
[9] "Estudio de perfiles de visitantes de un website a partir de los logs de los servidores web aplicando técnicas de Data Mining (webmining)" O. Marbán, http://www.ls.fi.upm.es/doctorado/Trabajos20012002/OMarban.doc, 2002
[10]"Sitios web adaptativos", Mike Perkowitz y Oren Etzioni, Dpto. Informática e Ingeniería Universidad de Washington, http://www.estadistico.com/arts.html?20020311, 2002
[11] "BIRCH: an efficient data clustering method for very large databases", Tian Zhang et al., International Conference on Management of Data, 1996

Buscando el Perfil de Usuario a partir de su Navegación en la Web

Arturo Peñarrubia, Antonio Fernández-Caballero y José M. Gascueña

Laboratory of User Interaction and Software Engineering
Computer Science Research Institute
University of Castilla-La Mancha, Albacete (Spain)
caballer@info-ab.uclm.es

Resumen. A lo largo del documento se plantea cómo llevar los principios de adaptatividad de interfaces de usuario al terreno de las aplicaciones web. Para ello necesitaremos ser capaces de establecer ciertas características de estos usuarios, características que serán empleadas en el proceso de adaptación. Basándonos en estas ideas, hemos implementado un portal web capaz de recopilar información de la sesión de navegación de los usuarios. Para ello, consultará los metadatos asociados a los diferentes documentos. A partir de esta información, la interfaz recomendará aquellas páginas del portal que se ajusten mejor a los intereses del usuario.

1 Introducción

El estudio de la interacción persona-ordenador está despertando en los últimos años un interés creciente desde diferentes perspectivas, y en particular desde la Ingeniería del Software [4][14][19]. La necesidad de facilitar el acceso a sistemas de información cada vez más complejos ha provocado un desarrollo sin precedentes en este campo de investigación. Y son precisamente las aplicaciones que hacen uso de una base de información tan compleja por su extensión y heterogeneidad como la web, las que requieren de un tratamiento muy importante en cuanto a esa interacción.

Si es necesario que un usuario que accede a un sistema de información sea capaz de averiguar fácilmente qué puede hacer con el mismo, y de realizar dichas tareas de una manera cómoda y eficaz.

Cuando hablamos de sistemas de cierta complejidad, ésta no es una tarea trivial [3][6][11][17]. Un usuario con ciertos requerimientos de información que inicie una sesión de navegación Web se encontrará con ciertas herramientas capaces de facilitarle aspectos muy concretos. Dispondrá por ejemplo de una serie de buscadores, capaces de proporcionarle, de una manera eficiente, un número importante de documentos que contengan ciertos términos, o incluso soportarán búsquedas mediante expresiones sencillas formadas en base a esos términos.

Muchos de ellos le permitirán también acceder a buena parte de esa información navegando por un árbol temático. Sin embargo, esas herramientas, ni consiguen facilitar la interacción entre el usuario y el sistema de información de una manera

global, ni solucionan los problemas concretos de ese usuario, al ofrecer soluciones demasiado genéricas.

2 Adaptatividad de Interfaces Web

2.1 Perfiles de Usuario

En los aspectos de interacción persona-ordenador se ha avanzado mucho en los últimos años, aunque siempre en relación a otros tipos de sistemas de información. Es el caso, por ejemplo del estudio de la adaptatividad de interfaces de usuario [7][12][13][18]. Evidentemente, si pretendemos que nuestra interfaz satisfaga a un grupo de usuarios demasiado amplio, reduciremos el grado de satisfacción de cada uno de ellos. Por tanto, para lograr interfaces más usables, es necesario conocer cómo son esos usuarios y de qué manera interaccionan con la interfaz. Es entonces cuando establecemos los denominados perfiles de usuario [2][11]. Estos perfiles nos permiten agrupar a los posibles usuarios en diferentes conjuntos, con una serie de características en común y, por tanto, con una serie de pautas de comportamiento en cuanto a su interacción. La inclusión de un usuario en uno de estos perfiles implicará la adaptación de la interfaz de la aplicación en todo momento a dicho perfil y, por tanto, a ciertas características de dicho usuario, facilitando de este modo su interacción. Pero, ¿podemos extender este planteamiento a un contexto como el de las aplicaciones web?

A lo largo de las distintas sesiones de navegación que pueda haber llevado a cabo, un usuario tenderá a seguir ciertas pautas en función de los requerimientos de información en cada instante. No es objeto de este trabajo determinar en qué manera evolucionan esas pautas, ni definirlas de una manera precisa. Simplemente partimos de la idea de que esas pautas existen y que, al menos, basándonos en el principio de coherencia, podemos concluir que existirá cierta relación entre los requerimientos de información de un usuario en un instante dado, y la información que ha recibido a lo largo de la sesión.

Si conocemos los intereses de un usuario hasta un momento determinado, quizás tengamos bastante información acerca de los intereses del mismo en ese instante. Esa es, por tanto, una información muy útil, siempre y cuando seamos capaces de filtrarla y aprovecharla para mejorar la navegación.

No pretendemos conocer cómo es cada uno de esos usuarios y acumular esa información para hacer uso de ella en el futuro, sino simplemente establecer perfiles anónimos de sesión. Se trataría de información no persistente usada en la adaptación de la interfaz a las necesidades del usuario.

El primer punto que debemos establecer es qué tipo de información debemos recopilar acerca de la sesión de navegación. Hoy en día, cualquier navegador incorpora mecanismos para facilitar el acceso a páginas visitadas recientemente, mediante ayuda en el acceso a la barra de direcciones, o listados de enlaces, por ejemplo. También ofrecen mediante marcadores soporte para el acceso a direcciones usadas frecuentemente. Sin embargo, nosotros pretendemos ir más allá, facilitando el acceso a contenidos accedidos recientemente. Este es un punto de partida ambicioso,

ya que no es fácil extraer información de tipo temático de los documentos Web, con las características que tienen en la actualidad.

2.2. Web Semántica y Metadatos

Para ello, necesitamos incorporar un nuevo planteamiento de la web, la denominada web semántica [15][20]. Esta nueva web plantea enriquecer la información contenida en la web tradicional con metainformación que describa dicha información base. Existen muchos proyectos en funcionamiento al respecto [23][27], pero a día de hoy, la gran mayoría de los documentos web carecen aún de metadatos descriptivos.

Muchos proyectos plantean sus propios conjuntos de etiquetas para los metadatos, pero uno de los más aceptados en la actualidad es Dublin Core [24]. Dublín Core nació como resultado del trabajo que viene desarrollando desde 1995 un grupo de expertos en la materia dirigido por Stuart Weibel. Plantea un conjunto de quince etiquetas, aunque actualmente existen partidarios de un conjunto ampliado. Las etiquetas tienen nombres como Title, Creator, Subject, Description, etc. (ver tabla 1).

Title	Publisher	Format	Relation
Creator	Contributor	Identifier	Coverage
Subject	Date	Source	Rights
Description	Type	Language	

Tabla. 1. Etiquetas Dublin Core

Es la tercera de estas etiquetas la que tiene un interés especial para el objetivo que perseguimos. Sin embargo, aún se nos siguen planteando muchos interrogantes. ¿Cómo interpretamos ese campo? ¿Es esta información por sí misma útil a nuestros propósitos?

2.3 Ontologías

Aparentemente no hemos avanzado mucho sobre el planteamiento de buscadores como Google [25]. Disponemos de los términos realmente relevantes acerca del contenido del documento, pero somos incapaces de interpretar el sentido de los mismos. Cuando un usuario realiza una búsqueda en Google sobre un término polisémico, el buscador retorna las páginas en las que aparece dicho término en cualquiera de sus sentidos.

Por otro lado el buscador obliga a que exista una concordancia léxica perfecta, ya que busca cadenas desconociendo su interpretación semántica. Sin embargo, la web semántica ya previene este tipo de conflictos. Para ello define el concepto de ontología [27][28]. Una ontología establece relaciones entre los diferentes términos aparecidos en la descripción de los documentos. Así pues, una ontología como RDF [29], establece un formato de sentencia <sujeto> <verbo> <predicado> en el que cada uno de estos campos estará expresado en la tripleta mediante la URI del documento que contiene la definición de ese elemento. Esto permite, por ejemplo, definir

equivalencias de términos entre otros tipos de relaciones. Es esta simplicidad la que hace de RDF una ontología con alto grado de flexibilidad.

Existen muchas otras ontologías en la actualidad, algunas de ellas ideadas para propósitos muy diferentes a la web semántica. Recientemente, un grupo de investigadores de diferentes universidades europeas ha desarrollado una base de datos léxica multilingüe denominada EuroWordNet [26] a partir de un proyecto anterior monolingüe denominado WordNet [22]. EuroWordNet está construido sobre una ontología mucho más rígida que RDF. La unidad mínima de información es el denominado synset. Un synset es un conjunto de palabras de la misma categoría gramatical (pueden ser nombres, verbos, adjetivos o adverbios) con el mismo significado en un contexto dado. A partir de estos elementos, y mediante un conjunto limitado de relaciones, se establece una red semántica. Algunas de estas relaciones son sinonimia, antonimia, hiponimia, meronimia, etc. (ver figura 1). Esta ontología proporciona la suficiente expresividad como para alcanzar nuestros objetivos y nos permite procesar la información de manera eficiente.

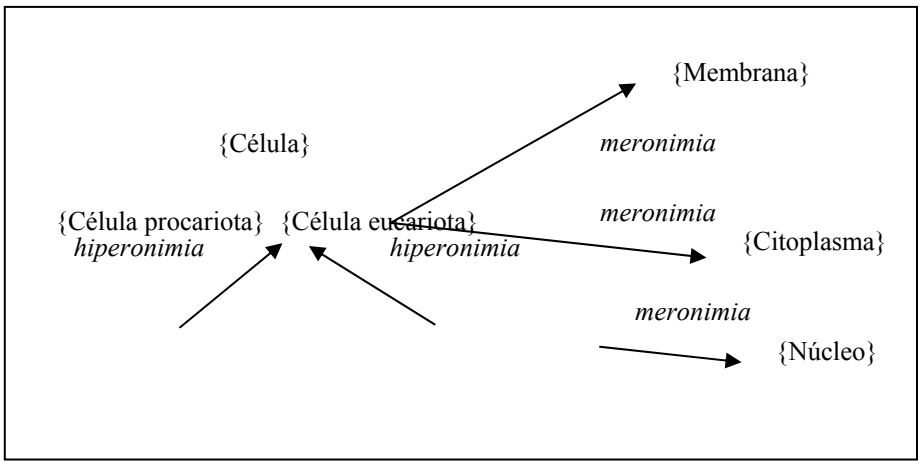


Fig. 1. Principales relaciones en EuroWordNet

Partiendo de una ontología como EuroWordNet seremos capaces de asociar páginas en las que aparecen términos relacionados semánticamente, sin que se trate necesariamente del mismo término.

Seremos capaces, por ejemplo en el contexto de la química orgánica de facilitar al usuario el acceso a páginas sobre hidrocarburos saturados donde no aparezcan con este nombre, sino con términos como parafina o alcano, que son equivalentes. Seremos capaces del mismo modo de discernir entre una página que trata sobre carabinas, en el sentido de transportador celular, de otras en las que aparezca como arma de fuego.

Una vez hemos establecido los formalismos sobre los que asentaremos nuestro proyecto, debemos especificar el uso que se hará de los mismos. El intento de llevar el concepto de adaptatividad de interfaces de usuario al contexto de las aplicaciones

web, podemos llevarlo a cabo a diferentes niveles. Para comenzar hemos planteado un prototipo sencillo

3 Descripción del Marco de Trabajo

3.1 Partiendo de la Navegación

Cuando un usuario entra en nuestro portal web, se encuentra con la pantalla dividida en dos frames. A la izquierda puede ver un frame menor en el que le iremos mostrando nuestras sugerencias, mientras que el resto de la pantalla lo ocupa el frame principal, en el que se cargarán las páginas que el usuario va visitando (ver figura 2). Nuestra sugerencia consistirá simplemente en un conjunto de enlaces a aquellas páginas relacionadas en cierto grado con las visitadas por el usuario.

No aspiramos a realizar búsquedas globales en Internet, sino simplemente trabajar en el contexto de un portal concreto. Esta idea sería útil, por ejemplo, en un portal dedicado a integración de diferentes administraciones públicas, o en portales con documentos de carácter científico, donde sea complicada su consulta por el número y heterogeneidad de los mismos.

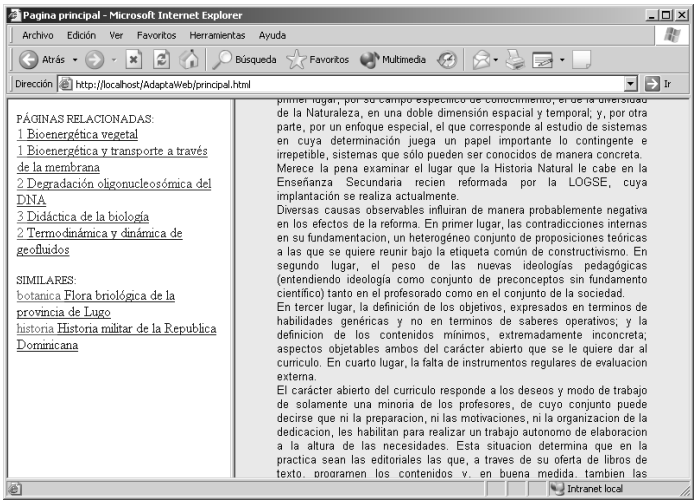


Fig. 2. Aspecto general del portal

Para ello, en el momento en que el usuario accede, se crea una cookie que almacenará los temas aparecidos en los documentos consultados, y la frecuencia de cada uno de ellos. Cuando el usuario accede a una página, actualiza la frecuencia de los temas en los que esta página coincide con su historial, e incorpora los nuevos. En cada momento el usuario puede ver un listado de páginas en las que tienen temas en

común con las visitadas. Para ello, el servidor realiza una búsqueda sobre los metadatos de los documentos disponibles en el portal. Podríamos haber incorporado las etiquetas DC en algunos documentos, como por ejemplo en páginas HTML, mediante XML, pero hemos optado por un tratamiento homogéneo para todos los posibles documentos, debiendo ir cada uno de ellos acompañado de su descripción en un fichero XML. Así pues, serán estos ficheros los consultados en el servidor para obtener el listado.

Sin embargo, no es la equivalencia léxica lo que buscamos, sino cierto grado de relación semántica. Para ello, disponemos de una base de información ontológica, consistente en una serie de ficheros de datos, y las correspondientes primitivas de consulta. El formato de los ficheros de datos es el utilizado en importación/exportación de synsets en EuroWordNet.

```
0 @9169@ WORD_MEANING
 1 PART_OF_SPEECH "n"
 1 VARIANTS
  2 LITERAL "celula eucariota"
   3 SENSE 1
   3 STATUS 99
 1 INTERNAL_LINKS
  2 RELATION "has_hyperonym"
   3 TARGET_CONCEPT
    4 PART_OF_SPEECH n
    4 LITERAL "celula"
     5 SENSE 1
  2 RELATION "has_mero_part"
   3 TARGET_CONCEPT
    4 PART_OF_SPEECH n
    4 LITERAL "membrana"
     5 SENSE 1
  2 RELATION "has_mero_part"
   3 TARGET_CONCEPT
    4 PART_OF_SPEECH n
    4 LITERAL "nucleo"
     5 SENSE 1
1 EQ_LINKS
```

Fig. 3. Fichero con información ontológica

Como podemos ver en la figura 3, estos ficheros se encuentran divididos en bloques de texto que contienen la descripción de cada uno de los synsets. En la cabecera del bloque encontramos los términos que constituyen el mismo, y a continuación aparece una lista de relaciones en las que participa ese synset. Los códigos numéricos que aparecen se emplean en EuroWordNet para relacionar esta información con los registros ILI, que asocian los synsets de los diferentes idiomas.

3.2 Manejando Información Ontológica

A partir de un conjunto de synsets estructurados de la forma que hemos visto, y por medio de una interfaz adecuada podemos obtener el soporte semántico que necesita nuestro modelo. En la tabla 2 podemos ver los métodos que constituyen esa interfaz

Función	Valor devuelto
Ret_dir_rel	Conjunto de términos relacionados con uno dado, por medio de una relación, en un sólo paso.
Ret_relationed	Conjunto de términos relacionados con uno dado, por medio de una relación (en cualquier número de pasos)
Return_relationed	Obtiene, a partir de un término y su sentido, todos aquellos relacionados a partir de una lista y un grado.
Is_relationed_s	¿Están dos términos relacionados por medio de una relación? (en cualquier número de pasos)
Has_common_ancestor	A partir de dos términos, sus sentidos, y una relación, determina si tienen un antecesor común
Synset	Conjunto de términos de un *synset*
Sense_set	Conjunto de sentidos de un término.
Unique_sense	¿Es un término monosémico?
Closeness	Función de proximidad.
Sense_of	Sentido de un término a partir de un contexto.

Tabla. 2. Funciones de consulta de información ontológica

En dicha tabla podemos ver los métodos que permiten manejar esta información ontológica. Los primeros de ellos nos sirven para navegar por el árbol semántico (ver figura 1) y obtener términos alcanzables mediante ramas de cualquier longitud, pero a través de una relación dada. Sin embargo, son las últimas funciones las que tienen un especial interés. Comentaremos algunas de ellas:

> Closeness: Cuantifica la proximidad semántica de dos términos. Para ello tiene en cuenta la distancia que separa sus nodos en el árbol y la relevancia de las relaciones que ha utilizado. Se utiliza a la hora de decidir cual de los sentidos de un término se ajusta más a un contexto (en cual es mayor esta función).
> Sense_of: Determina el sentido de un término en función de un contexto (una lista de términos de los que desconocemos su sentido). Es por tanto esta función la que nos permite pasar de la unidad léxica con la que trabajamos a nivel de metadatos a la unidad semántica necesaria en el acceso a la ontología.

Este paso es necesario si pretendemos que las dos fases sean independientes, es decir, que cuando se da de alta un documento, y se escribe su temática, no se tenga que conocer la ontología. La información temporal que almacenamos acerca de la

sesión, también es de carácter semántico. Por tanto, para contrastar nuevas páginas con el historial debemos realizar una conversión.

Recordemos que existe una escala de proximidad entre términos. Así pues, tendremos proximidad en grado 0 cuando dos términos coinciden léxicamente, grado 1 si son sinónimos, grado 2 si están separados por un paso en alguna de las principales relaciones... Este mecanismo nos hace posible configurar el portal para trabajar con diferente precisión, ya que las correspondientes funciones están parametrizadas.

Esta interfaz nos permite utilizar nuestra ontología para interpretar los metadatos asociados al documento, para extraer la información semántica de los mismos. En ocasiones, un metadato aislado puede presentar aún ambigüedad semántica, pero es entonces cuando nuestra ontología nos permite resolverla a través del contexto.

3.3 Usando la Información Obtenida

Como decíamos, es el servidor el que puede explorar las características de los documentos presentes en el portal. De cada documento nos interesa obtener el conjunto de temas que lo describen. A partir de esa lista de términos, y con las herramientas anteriormente estudiadas, será capaz de determinar en qué medida puede interesar esa página al usuario.

Podríamos decir que tenemos una navegación asistida por agentes software [1][21]. Y es que, desde esa perspectiva, es un autómata el encargado de mantener una lista actualizada de enlaces que sirven al usuario para conseguir sus objetivos de manera más cómoda y eficiente.

Si la página coincide en algunos de los temas de interés para el usuario en la sesión, deberemos proponérsela, mostrándosela en el frame izquierdo. En función del grado de coincidencia, deberemos priorizar un enlace, incluso mostrando sólo los de mayor interés.

Disponemos de mecanismos para cuantificar la proximidad de la página a los diferentes temas consultados por el usuario, pero tal vez esos temas no sean igualmente relevantes. Debemos ser capaces de estudiar en cierto modo la evolución de la conducta del usuario, ya que toda la información recopilada en la sesión no es igualmente útil en un momento dado.

3.4 Obteniendo Patrones de Comportamiento

Probablemente, no tendrá el mismo valor para el usuario un documento en el que se trate el mismo tema que en el documento que está leyendo, que otro documento que trate temas similares. Por tanto, debemos primar esa concordancia exacta (o mediante sinónimos). Esto nos obliga a mantener el historial de términos consultados lo más completo posible, en vez de recurrir a almacenar, por ejemplo, términos significativos de familias semánticas que harían más eficiente el uso de ese historial. Sin embargo, debemos ir más allá.

Comentábamos anteriormente que sería útil almacenar el número de veces que aparece un tema. Evidentemente un tema consultado frecuentemente por el usuario debe ser priorizado. Los mecanismos usados para la captura del interés de un usuario

para un término dado del léxico son los denominados de computación acumulativa [5][16].

$$pal(t) = (1-\lambda^2)\cdot F_p\left[pal(t-\Delta t), ent(t)\right] + \\ + \frac{\lambda(\lambda+1)}{2} pal_{\min} + \frac{\lambda(\lambda-1)}{2} pal_{\max},$$

$$\lambda = \begin{cases} 1, & si \quad F_p[...] \geq pal_{\max} \\ 0, & si \ pal_{\min} < F_p[...] < pal_{\max} \\ -1, & si \ pal_{\min} \geq F_p[..] \end{cases}$$

El interés sobre el término pal es función del interés en el instante de tiempo anterior y de la entrada en el instante actual y está acotado entre dos valores, el mínimo palmin y el máximo palmax posible de interés.

Este proceso se complica en cuanto contemplamos algunos aspectos más. Debemos tener en cuenta que nuestro modelo no está basado en elementos aislados, sino que el uso de la ontología nos permite trabajar con redes semánticas. Por tanto, no podemos despreciar la influencia que puede tener la consulta de un nodo en los nodos relacionados con éste. Para ello, se usan mecanismos de interacción lateral [10], en los que los términos relacionados se ven influidos por la aparición de otros.

Mediante la conjunción de ambos mecanismos, interacción lateral en computación acumulativa [8][9], la aparición de un término debería aumentar la prioridad de otros términos relacionados, en mayor medida cuanto mayor sea el grado de proximidad, siendo el máximo incremento para el propio término.

Por otra parte, el usuario puede consultar a lo largo de la sesión diferentes áreas temáticas. El interés de cada una de ellas también debería verse reflejado. De modo que se van creando diferentes patrones de comportamiento obtenidos en diferentes áreas de la ontología global.

Pero no solo debemos contemplar la inferencia positiva. Es posible que esa prioridad pueda ser también decrementada, opción que también está contemplada e los mecanismos de computación acumulativa. Tal vez relaciones como la antonimia deban tener efectos negativos sobre un término. Sería interesante también, que una consulta reciente no tenga el mismo valor que otra al principio de la sesión.

Estos y otros factores deben ser tenidos en cuenta en un intento de acercarse a posibles evoluciones de los intereses del usuario. Sin embargo, como dijimos al comienzo del presente documento, aproximarse realmente a esa evolución es una tarea compleja, que requiere un estudio desde ámbitos diferentes a los aquí perseguidos.

4 Una Sesión de Navegación

Hasta ahora hemos planteado los formalismos en los que se basa el proyecto, y hemos comentado en líneas generales cómo hacer uso de los mismos en el prototipo. Para ofrecer una visión integrada de las ideas hasta ahora formuladas, a continuación abordaremos la exposición de un caso de estudio concreto. Se trata de un ejemplo sencillo que nos permitirá estudiar la manera en que se han aplicado estas ideas en el prototipo.

En primer lugar, definiremos parcialmente la ontología usada por el sistema. Nos limitaremos a exponer la parte del esquema necesaria para la comprensión del ejemplo. El esquema de la figura 4 nos expone esta información.

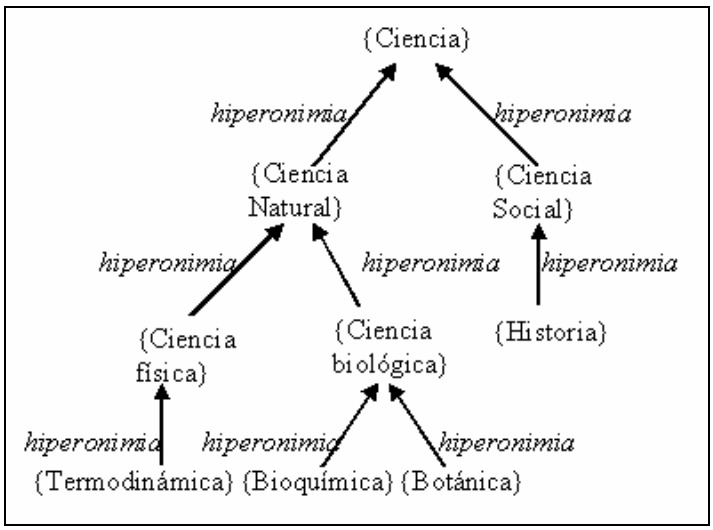


Fig. 4. Información ontológica en el caso de estudio

Una vez hemos definido la red de términos, podemos ver cómo las diferentes páginas del portal hacen uso de los mismos. En la tabla 3 podemos ver los campos Title y Subject de la descripción Dublin Core de cada uno de los documentos.

Tras leer las sugerencias, el usuario decide visitar la página 7 titulada "Bioenergética y transporte a través de la membrana" (ver figura 6). Una vez cargada la página, la interfaz tendrá el aspecto mostrado en la figura 6.

Como vemos en la figura 6, se le recomienda como página similar la que acaba de visitar. Antes del nombre de la página aparece el número de coincidencias de esa página con su historial, en este caso 2, ya que todos los metadatos de esta página (termodinámica y geofluido) están en el historial al haber sido visitada justo antes.

Nuevamente, el usuario decide visitar una de las páginas recomendadas, en este caso la Nº 6 titulada "Degradación oligonucleosómica del DNA" (ver figura 7).

	Title	**Subject**
1	Didáctica de la biología	didáctica, biología, enseñanza
2	Historia militar moderna de la Republica Dominicana	historia, carabina, Republica Dominicana, ejército
3	Termodinámica y dinámica de geofluidos	termodinámica, geofluido
4	Bioenergética vegetal	bioenergética, fotosíntesis, cloroplasto, bioquímica
5	Flora biológica de la provincia de Lugo	botánica, flora, corología, Lugo
6	Degradación oligonucleosómica del DNA	DNA, carabina, membrana, bioquímica
7	Bioenergética y transporte a través de la membrana	Bioenergética, termodinámica, membrana

Tabla. 3. Metadatos del caso de estudio

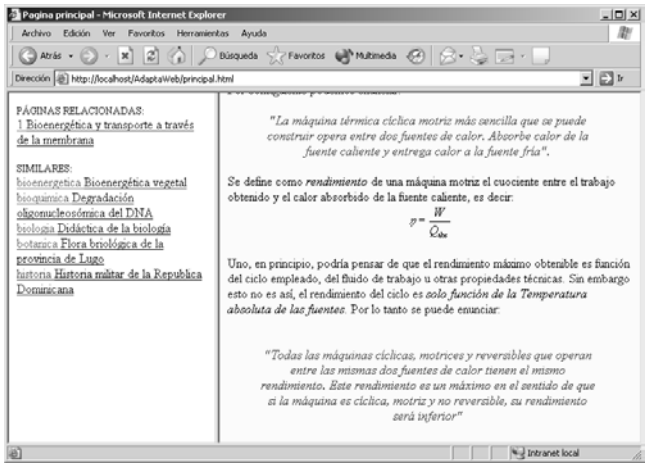


Fig. 5. El usuario visita el documento Nº 3

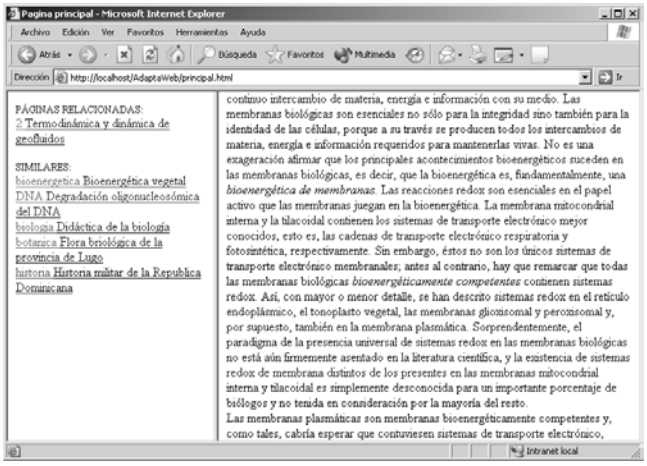


Fig. 6. El usuario visita el documento Nº 7

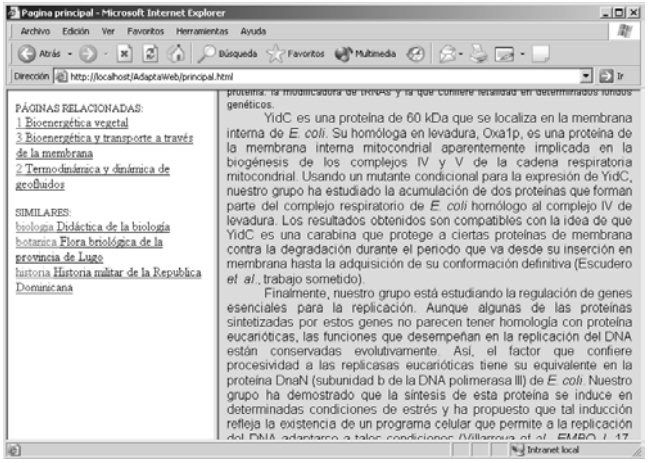


Fig. 7. El usuario visita el documento Nº 6

Como vemos, entre las páginas relacionadas se encuentra, además de las visitadas, la Nº 4, por tratar sobre bioenergética, como la página en la que se encuentra el usuario, aunque desde diferentes enfoques.

Conclusiones

Al principio de este documento, se planteaba llevar la adaptatividad de interfaces de usuario al terreno de las aplicaciones web. Decíamos, que ésta no era una tarea sencilla por las características de este tipo de aplicaciones.

Por esta razón cuando comenzábamos a abordar la puesta en práctica de esa idea mediante un prototipo, asumíamos ciertas restricciones. Restricciones como las asociadas al volumen de documentos accesibles (nos limitamos a nuestro portal web), o la complejidad del perfil de usuario (simplemente el conjunto de temas visitados) nos permitían llevar a cabo una sesión de navegación inteligente asistida por una interfaz adaptativa.

Sin duda es mucho el trabajo por realizar en nuestro intento de conseguir que la información que suministra en cada momento la interfaz al usuario sea lo más útil posible para el mismo. Hablábamos en el punto anterior, por ejemplo, de la necesidad de filtrar los documentos que más se aproximen al perfil del usuario, y con anterioridad, comentábamos también la posibilidad de primar la información más reciente, así como de controlar la interacción lateral en la computación acumulativa. Estas y otras mejoras serán estudiadas sin duda en un futuro. El planteamiento de este primer prototipo, nos ha permitido, como esperábamos, estudiar las características del modelo, sus carencias, y el grado de aproximación a las intenciones iniciales.

Agradecimientos

Este trabajo está financiado en parte por el proyecto de la Junta de Comunidades de Castilla-La Mancha PBC-03-003.

Referencias

[1] Bradshaw, J. M. (Editor). Software Agents. The MIT Press, 1997.
[2] Chan, P. A non-invasive learning approach to building web user profiles. Proceedings of the KDD-99 Workshop on Web Analysis and User Profiling. Florida Institute of Technology, Melbourne, Australia.
[3] Cueva, J.M., González, B.M., Aguilar, L., Labra, J.E. y del Puerto, M. (Editores). Proceedings of the International Conference on Web Engineering, ICWE 2003, Lecture Notes in Computer Science 2722, 2003.
[4] Dix, A., Finlay, J., Abowd, G. y Beale R. Human-Computer Interaction. Prentice-Hall, 1998.
[5] Fernández-Caballero, A. Modelos de interacción lateral en computación acumulativa para la obtención de siluetas. Tesis doctoral, Universidad Nacional de Educación a Distancia (España) 2001.
[6] Fernández-Caballero, A., López-Jaquero, V., Montero F. y González, P. Adaptive interaction multi-agent systems in e-learning / e-teaching on the web. Third International Conference on Web Engineering, ICWE 2003, Lecture Notes in Computer Science 2722, pp. 144-153, Springer-Verlag.
[7] Fernández-Caballero, A., López-Jaquero, V. y Montero F. Métricas de usabilidad y sistemas multiagente en hipermedia adaptativa. Interacción 2003, Vigo (España), 2003.
[8] Fernández-Caballero, A., Fernández, M.A., Mira J. y Delgado A.E. Spatio-temporal shape building from image sequences using lateral interaction in accumulative computation. Pattern Recognition, 36(5), pp. 1131-1142, 2003

[9] Fernández-Caballero, A., Mira, J., Delgado, A.E. y Fernández, M.A. Lateral interaction in accumulative computation: A model for motion detection. Neurocomputing, 50C, pp. 341-364, 2003.
[10] Fernández-Caballero, A., Mira, J., Fernández, M.A. y López, M.T. Segmentation from motion of non-rigid objects by neuronal lateral interaction. Pattern Recognition Letters, 22 (14), pp. 1517-1524, 2001.
[11] Fons, J., García, F. J., Pelechano, V. y Pastor, O. User Profiling Capabilities in OOWS. Proceedings of the International Conference on Web Engineering, ICWE 2003, Lecture Notes in Computer Science 2722, pp. 486-496, 2003
[12] López-Jaquero, V., Montero, F., Molina, J.P., Fernández-Caballero, A. y González, P. Model based design of adaptive user interfaces through connectors. 10th Workshop on Design, Specification and Verification of Interactive Systems, DSV-IS 2003, Lecture Notes in Computer Science 2844, 2003.
[13] López-Jaquero, V., Montero, F., Fernández-Caballero, A. y Lozano, M.D. Towards adaptive user interface generation: One step closer to people. Enterprise Information Systems V, Kluwer Academia, 2003.
[14] Lorés, J., Abascal, J., Cañas, J.J., Gea, M. Gil, A.B., Lorés, J., Martínez, A.B., Ortega, M. Valero, P. y Vélez, M. (Editores). La Interacción Persona-Ordenador. AIPO, Asociación Interacción Persona Ordenador 2001, 2001.
[15] Macías, J. A. y Castells, P. Tailoring Dynamic Ontology-Driven Web Documents by Demonstration. Proceedings of the 6th International Conference on Information Visualisation - International Symposium of Visualisation of the Semantic Web. IEEE Computer Society, 2002.
[16] Mira, J., Fernández, M.A., López, M.T., Delgado, A.E. y Fernández-Caballero, A. A model of neural inspiration for local accumulative computation. 9th International Conference on Computer Aided Systems Theory, EUROCAST 2003, Lecture Notes in Computer Science 2809, pp. 427-435, 2003.
[17] Olsina, L., de los Angeles M., Fons, J., Mara S., Pastor, O. Towards the Design of a Metrics Cataloging System by Exploiting Conceptual and Semantic Web Approaches. Proceedings of the International Conference on Web Engineering, ICWE 2003, Lecture Notes in Computer Science 2722, pp. 324-333, 2003.
[18] Ortigosa, A. y Carro, R. Asistiendo el Proceso de Mejora Continua de Cursos Adaptativos. III Jornadas de Interacción Persona-Ordenador, INTERACCIÓN-2002 Leganés, 8-10 de Mayo 2002.
[19] Ramos, I., Fernández-Caballero, A. y Lozano, M.D. Tendencias Actuales en la Interacción Persona-Ordenador: Accesibilidad, Adaptabilidad y Nuevos Paradigmas. XIII Escuela de Verano de Informática, Universidad de Castilla-La Mancha, Departamento de Informática, Albacete (España), 2003.
[20] Wahlster, W., Lieberman, H. y Hendler, J.A., Spinning the Semantic Web: Bringing the World Wide Web to its Full Potential. The MIT Press 2003.
[21] Weiss, G. (Editor). Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, The MIT Press, 2000.
[22] WordNet: A Lexical Database for the English Language. http://www.cogsci.princeton.edu/~wn
[23] The Maryland Information and Network Dynamics Lab Semantic Web Agents Project. http://www.mindswap.org/
[24] Dublin Core Metadata Initiative. http://www.dublincore.org
[25] Google. http://www.google.com
[26] EuroWordNet Consortium. http://www.illc.uva.nl/EuroWordNet
[27] On-To-Knowledge: Content-driven Knowledge-Management through Evolving Ontologies. http://www.ontoknowledge.org

[28] The Semantic Web Community Portal, Markup Language and Ontologies. http://www.semanticweb.org/knowmarkup.html
[29] Resource Description Framework (RDF). http://www.w3.org/RDF