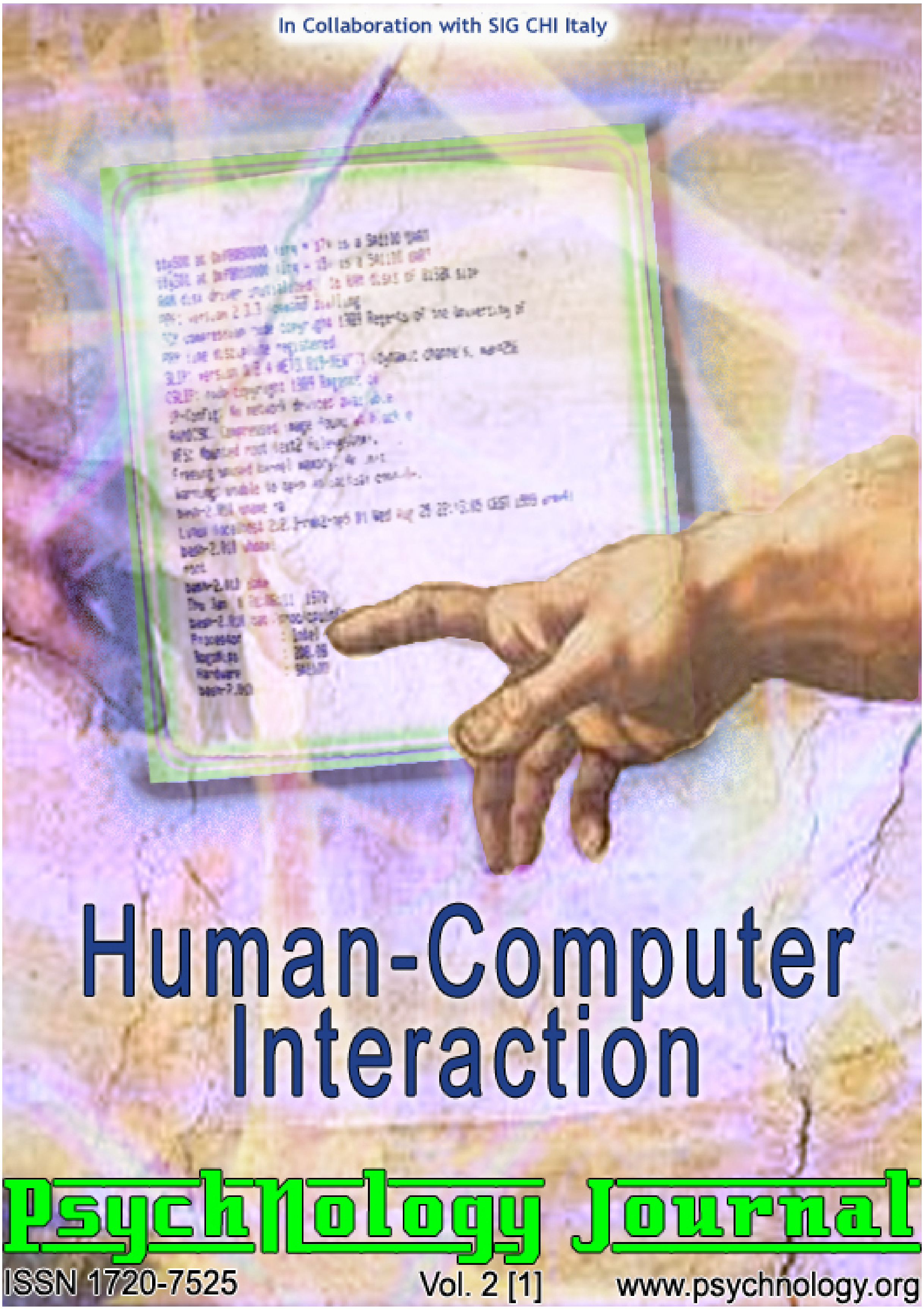

In Collaboration with SIG CHI Italy
Human-Computer
Interaction
PsychNology Journal
ISSN 1720-7525
Vol. 2 [1]
www.psychnology.org

TABLE OF CONTENTS

Editorial Preface

PsychNology Journal has now arrived to its fifth issue, particularly rich with topics and approaches and entirely devoted to Human-Computer Interaction. After dealing in the previous issues with new kinds of interfaces (Special Issue on Future Interfaces, Vol.1,n.3/4), the Journal broadens its scope now to the whole field of human interaction with technologies, a cherished theme by people interested in the ‘other side of technology’. The issue has benefited from the collaboration with SIG-CHI Italy and collects selected papers from the ‘HCItaly Conference 2003’, including the prestigious contribution from Peter Brusilovsky (University of Pittsburgh), invited speaker in the Italian event and invited author here.

As Editors-in-Chief of PsychNology, we warmly thank Liliana Ardissono and Anna Goy, brilliant scholars at the Department of Computer Science (Turin, Italy), chair persons of HCITALY 2003 and flawless guest editors of this issue of PsychNology.

Luciano Gamberini
Anna Spagnolli

Editors-in-Chief

Introduction to the Special Issue

The recent expansion of the Internet has opened the avenue to services supporting the access to remote data repositories, the search for information and documents, and the execution of various everyday activities, such as home banking and e-government. Moreover, the research about intelligent systems has enabled the human user to remotely pilot smart machines performing difficult or dangerous tasks.
Unfortunately, the development of user interfaces supporting the interaction with complex services and devices has not kept the pace with the technological development, so that the potential offered by the technology is challenged by the limited usability of the new products and services. The gap between the user-oriented view of the offered service and the technical level concerning its implementation has to be filled in order to make machines and software systems acceptable to the end-user. Moreover, the user's trust in automated systems has to be enhanced, by taking predictability and controllability issues into account.
The design and development of usable user interfaces has attracted the attention of various research communities. For instance, the Human Computer Interaction research has focused on the definition of methodologies for the development of user interfaces that meet the user's interaction requirements during the life cycle of products and services. Moreover, the User Modeling and Intelligent User Interfaces research has focused on the definition of techniques for the dynamic adaptation of services to the individual user's preferences. Furthermore, within the Adaptive Hypermedia field, researchers have developed Web-based systems that tailor the content, the interaction and the layout to the needs of users having heterogeneous knowledge, interests, and exploiting different devices to interact with the services (e.g., smart phones, PDAs, portable and desktop computers).

Fundamental challenges that must be addressed in order to make products and services widely usable include:

- The adaptation of User Interfaces to the user's device and to multiple communication channels, as well as to bandwidth and other related constraints.
- The mediation between the user's interaction requirements and the technical aspects underlying the service execution.
- The enhancement of the usability of automated decision makers and the satisfaction of user's requirements concerning the predictability and controllability of their behavior.
- The support for ubiquitous and multimodal interaction with the user, taking not only technical issues, but also accessibility and pleasure requirements into account.
- The availability of infrastructures facilitating the development of multi-platform services.
- The provision of adaptive features and direct support to the user during the exploration of virtual environments.
- The availability of user-centered software development methodologies supporting the implementation of systems that can be easily configured and tailored to the user's needs.
- The definition of guidelines and strategies for the design of usable multimodal and multi-platform user interfaces, as well as the availability of software environments supporting the system development.

The Italian Intelligent User Interfaces (IUI) and Human Computer Interaction (HCI) communities have proposed leading solutions to address the above-mentioned challenges, which have been presented in several International Conferences and Journals. In particular, this special issue includes a selection of the work presented at HCITALY 2003, the 3rd Human-Computer Interaction Symposium of SIG-CHI Italy, organized by the Computer Science Department of the University of Torino (Italy). The papers are focused on 5 main themes: adaptivity and user support in 3D and animated user interfaces, user interface design issues, end-user software development and tailoring, infrastructures for the development of multi-platform interactive applications, and user interfaces supporting the interaction with automated problem solvers. Specifically:

- Peter Brusilovsky, invited speaker at the Symposium and author of the first article, gives a brief history of adaptive navigation support in the pre-Web Hypermedia and discusses challenges emerging with the evolution from Adaptive Hypermedia to the Adaptive Web. The paper also introduces the concept of adaptive support to the exploration of Virtual Reality environments and presents the results of a number of projects, carried out at the University of Pittsburgh, focused on the study of adaptive navigation support in different contexts. The paper includes three videos demonstrating the employment of adaptive navigation techniques in a virtual museum.

- The adaptive navigation of 3D environments is also the focus of the paper by Chittaro et al., who propose the introduction of embodied characters guiding the exploration of the environments as an effective aid to the user. In particular, the authors propose a tool supporting the automated code generation for adding guided tours to virtual environments and they describe the deployment of the proposed tool in a virtual museum.

- Abbattista et al. propose a different perspective on embodied characters: in the paper, they overview the exploitation of animated agents as user interface components of Web-based services. Moreover, they describe the conversational and emotional features offered by SAMIR, a 3D conversational agent recommending books in a Web-based store.

- The paper by Valitutti et al. addresses open issues in the development of expressive user interface agents having advanced conversational capabilities. The authors discuss the importance of Affective Computing in HCI, where the ability of displaying emotions and of understanding the user's emotional state plays a key role. Moreover, they present WORDNET-AFFECT, an affective lexicon that correlates affective concepts to words and represents a basic resource for the development of emotional user interfaces.

- Di Nocera et al. focus on Web page design and describes experiments aimed at suggesting criteria for the organization of objects that optimize the user performance. They hypothesize that the location of some web objects, such as links to specific contents, are expected by the users at specific spatial locations, and proposes the Cognitive GeoConcept procedure for supporting the information architect's decisions.

- In the paper by Costabile et al., the user participation in the configuration and adaptation of software to individual needs is proposed as a mean to address the usability issues emerging during the life cycle of software. The authors present the Software Shaping Workshops environment, which supports the development and customization of software enabling both software engineers and domain experts to modify products to satisfy their emerging needs and requirements.

- Chesta et al. present a model-based approach to the development of context-sensitive, multi-platform, nomadic applications. In the paper, the authors describe the TERESA tool, which supports the development activity by providing different levels of abstraction for the specification of device-dependent user interfaces and the enforcement of the consistency between the various interfaces.

- The article by Cortellessa et al. addresses usability and trust issues concerning the interaction with automated decision makers. The authors present an intelligent user interface mediating the interaction between the user and an automated problem solver that schedules tasks during space missions. Moreover, they present the results of a preliminary evaluation of the system aimed at assessing the extent to which the proposed user interface features enhance the usability and acceptability of the system in real cases.

The HCITALY 2003 Symposium has been promoted by SIG-CHI Italy (http://giove.cnuce.cnr.it/sigchi/), the Italian association on Human-Computer Interaction. This association, run entirely by volunteers, is active since 1995 and has the goal of promoting and increasing knowledge and interest in the science, technology, design, development, and application of methods, tools, and techniques for Human Computer Interaction. SIG-CHI Italy has scientific and educational goals, and aims at providing a mean of communication between people interested in HCI; the hcitaly@cnuce.cnr.it mailing list is devoted to the exchange of information in the field. The association also organizes meetings, conferences, discussion groups and workshops. The main event is HCITALY, which provides a unique opportunity for the Italian HCI community to meet and discuss research results. In addition, daily events on specific topics with multi-disciplinary participation are organized. For example, the Usability and Accessibility day (http://giove.cnuce.cnr.it/usac.htm) has been recently organized at ISTI-CNR in Pisa. Moreover, the Natural Interaction day (http://naturalinteraction.org/workshop/) has been organized in Firenze by the local University. Furthermore, INTERACT (www.interact2005.org), the main IFIP HCI event, will take place in Roma on September 12-17, 2005. The next edition of HCITALY is planned in conjunction with this event.

Liliana Ardissono
Università di Torino
Guest Editor

Anna Goy
Università di Torino
Guest Editor

Fabio Paternò
ISTI-CNR Pisa

PsychNology Journal, 2004
Volume 2, Number 1, 7 - 23

Adaptive Navigation Support: From Adaptive Hypermedia to the Adaptive Web and Beyond

Peter Brusilovsky♣

School of Information Sciences
University of Pittsburgh

ABSTRACT

Adaptive navigation support is a specific group of technologies that support user navigation in "virtual spaces" adapting to the goals, preferences and knowledge of the individual user. These technologies, originally developed in the field of adaptive hypermedia, are becoming increasingly important in several adaptive Web applications from Web-based adaptive hypermedia to adaptive virtual reality. This paper provides a brief introduction to adaptive navigation support, reviews major adaptive navigation support technologies, and presents a sequence of projects performed by our group to study adaptive navigation support in different contexts.

Keywords: *Navigation support, user model, virtual environments, adaptive system, personalization.*

Received 15 March 2004; accepted 22 March 2004.

1. Introduction

Adaptive hypermedia (Brusilovsky, 2001) is a research area at the crossroads of hypermedia and user modeling. Adaptive hypermedia systems (AHS) offer an alternative to the traditional “one-size-fits-all” hypermedia and Web systems by adapting to the goals, interests, and knowledge of individual users represented in the individual *user models*. This paper is focused on *adaptive navigation support* technologies originally developed in the field of adaptive hypermedia. By adaptively altering the appearance of links on every browsed page using such methods as *direct guidance*, *adaptive ordering*, *link hiding and removal*, and *adaptive link annotation*, these technologies support personalized access to information. Adaptive navigation support technologies have been evaluated in several application areas and have

♣ Corresponding Author:
Peter Brusilovsky,
School of Information Sciences,
University of Pittsburgh
Pittsburgh PA 15260
Tel.: 1 (412) 624 9404
E-mail: peterb@mail.sis.pitt.edu

demonstrated their ability to let the users achieve their goals faster, reduce navigation overhead, and increase satisfaction (Brusilovsky, 1997).

Nowadays, adaptive navigation support technologies have been growing in importance in areas past the horizon of classic hypertext, their original application area. These technologies are now being used in several adaptive Web (Brusilovsky & Maybury, 2002) applications from Web-based adaptive hypermedia to adaptive virtual reality. This paper provides a brief introduction to adaptive navigation support, reviews main adaptive navigation support technologies and presents a sequence of projects performed by our group to study adaptive navigation support in different contexts.

2. Adaptive Navigation Support in pre-Web Hypermedia

The research on adaptive navigation support in hypermedia can be traced back to the early 1990's. At that time, several research teams had recognized the problems of static hypertext in different application areas, and had begun to explore various ways to adapt the behavior of hypertext and hypermedia systems to individual users. A number of teams addressed the problems related to navigation in hypermedia such as the problem of inefficient navigation or the problem of being lost that had been discovered when the field of hypertext reached relative maturity at the end of the 1980's (Hammond, 1989). Within a few years, a number of navigation support technologies were proposed (Böcker, Hohl & Schwab, 1990; Brusilovsky, Pesin & Zyryanov, 1993; de La Passardiere & Dufresne, 1992; Kaplan, Fenwick & Chen, 1993). While the proposed technologies were relatively different, they shared the same core idea: adapt the presentation of links located on a hypertext page (hypernode) to the goals, knowledge, and preferences of the individual user. The adaptive navigation support technologies introduced by early adaptive hypermedia systems were later classified as *direct guidance*, *sorting*, *hiding*, and *annotation* (Brusilovsky, 1996).

Direct guidance is the simplest technology of adaptive navigation support. Direct guidance suggests the "next best" node for the user to visit according user's goals, knowledge, or/and other parameters represented in the user model. To provide direct guidance, an adaptive educational hypermedia system usually presents an additional dynamic link (usually called "next" or "teach me") which is connected to the "next best" node, as illustrated, in ISIS-Tutor (Brusilovsky & Pesin, 1994), SHIVA (Zeiliger, 1993), and HyperTutor (Pérez, Gutiérrez & Lopistéguy, 1995). Direct guidance is very often applied in adaptive educational hypermedia systems that have roots in Intelligent Tutoring. In this group of systems, direct guidance is nothing else but hypermedia access to traditional curriculum sequencing mechanisms (Brusilovsky, 1992). A problem with direct guidance is that it provides no support for the users who don't wish to follow the system's suggestions. Direct guidance is useful but it should be used in conjunction with one of the "more supportive" technologies that are listed below.

The idea of *adaptive sorting* technology is to order all the links of a particular page according to the user model and some user-valuable criteria: the closer to the top, the more relevant the link is. Adaptive sorting was introduced in two early systems - Hypadapter (Böcker et al., 1990) and HYPERFLEX (Kaplan et al., 1993), however, it has not become very popular because of its limited applicability. It can be used with non-contextual links, but it can rarely be used for indexes and content pages (which usually have a stable order of links), and can never be used with contextual links and maps. Another problem with adaptive ordering is that this technology makes the order of links non-stable: it may change each time the user enters the page. For both reasons this technology is presently most often used for showing new links to the user in conjunction with link generation. The study of the HYPERFLEX system (Kaplan et al., 1993) showed that adaptive sorting can significantly reduce navigation time in search-oriented hypermedia applications.

The purpose of navigation support by *hiding* is to restrict the navigation space by hiding, removing, or disabling links to irrelevant pages. A page can be considered irrelevant for several reasons: for example, if it is not related to the user's current learning goal or if it presents materials which the user is not yet prepared to understand. Hiding protects users from the complexity of the whole hyperspace and reduces their cognitive overload. Educational hypermedia systems were the main application area where adaptive hiding techniques were suggested and explored. Indeed, beginning with just a part of the whole picture then introducing other components step by step as the student progresses through the course is a popular educational approach and adaptive hiding offers a simple way to implement this. Early adaptive hypermedia systems used a very simple method of hiding links - essentially removing the link together with the anchor from a page. A good example can be provided by the ISIS-Tutor system (Brusilovsky & Pesin, 1998) which made more and more links in an educational hypermedia visible following the growth of the student's knowledge of the subject (Figure 1). De Bra and Calvi (1998) later called the ISIS-Tutor approach *link removal* and have suggested and implemented several other variants for link hiding. In particular, link hiding and disabling became more popular since they leave the anchor (hot word) intact and just disable or hide the link itself. A number of studies of link hiding demonstrated that this is a "unidirectional" technology. While gradual link enabling as used in ISIS-Tutor was acceptable and effective, the reverse approach was found questionable: users become very unhappy when previously available links become invisible or disabled.

Fig. 1: Adaptive navigation support in ISIS-Tutor. The picture above provides an example of link annotation: the green color annotates links to known information, the red color annotates links to ready-to-learn information, and the light blue color annotates links to not-ready information. The picture below presents the same page now featuring a combination of link annotation and hiding. Links to not-ready information (shown as blue on the picture above) are removed.

The idea of *adaptive annotation* technology is to augment the links with some form of annotation, which can tell the user more about the current state of the nodes behind the annotated links. These annotations are most often provided in the form of visual cues. For example, Manuel Excel (de La Passardiere & Dufresne, 1992) associated links with different icons, ISIS-Tutor (Brusilovsky & Pesin, 1994) changed the color of the links (Figures 1), and Hypadapter (Hohl, Böcker & Gunzenhäuser, 1996) altered font sizes. Annotation can be naturally used with all possible forms of links. This technology supports a stable order of links and avoids problems with incorrect mental maps. Annotation is generally a more powerful technology than hiding: hiding can distinguish only two states for the related nodes - relevant and non-relevant - while existing applications of annotation can distinguish up to six states. For all the above reasons, adaptive annotation later grew into the most often used adaptive annotation technology.

Several early works have explored the value of adaptive navigation support. In the first published study, de La Passardiere & Dufresne (1992) conducted experiments with MANUEL EXCEL, providing the first evidence in favor of adaptive navigation support. A year after that, Kaplan et al. (1993) reported two studies of adaptive navigation support with their system HYPERFLEX, demonstrating that sorting-based adaptive navigation support can improve user performance in information search tasks. Our own exploration of two adaptive navigation support technologies, hiding and annotation as they were implemented in ISIS-Tutor system also delivered encouraging results (Brusilovsky & Pesin, 1998). We have compared three versions of the ISIS-Tutor: a non-adaptive version, a version with adaptive annotation, and a version with both hiding and annotation. The results of our study have demonstrated that the same educational goal can be achieved in either of the adaptive version with much less navigational overhead. The overall number of navigation steps, the number of unforced repetitions of previously studied concepts, and the number of task repetitions (i.e., trials to solve a previously visited task) were significantly smaller for both adaptive versions.

3. Adaptive Navigation Support in Web-based Hypermedia

The Web as "hypermedia for everyone" immediately provided an attractive platform for adaptive hypermedia applications. The majority of work on Web-based adaptive hypermedia has focused on exploring the original adaptive hypermedia technologies, but in the Web context. The work on pre-Web adaptive hypermedia provided a good foundation for the new generation of research. As the Web developed, the focus of work has also moved from exploring isolated techniques using "lab-level" systems to developing and exploring "real world" systems for different application areas such as E-learning, E-commerce, virtual museums, etc.. A good review of this generation of adaptive hypermedia systems was provided in (Brusilovsky, 2001).

Our own experience with the ISIS-Tutor system helped us to develop ELM-ART (Brusilovsky, Schwarz & Weber, 1996), the first practical Web-based system that used adaptive navigation support. ELM-ART has integrated a number of innovative techniques in a versatile adaptive Web-based course for the programming language

LISP, which have been used by hundreds of students over several years. In particular, ELM-ART has pioneered the idea of an adaptive electronic textbook and introduced the traffic light metaphor for adaptive navigation support in educational hypermedia. With this metaphor, green bullet in front of a link indicates recommended readings, while a red bullet indicates that the student might not have enough knowledge to understand the information behind the link. Other colors like yellow or white indicate other educational states such as the lack of new knowledge behind the link. Figures 2 shows adaptive annotation in its most recent versions of ELM-ART (Weber & Brusilovsky, 2001). A study of ELM-ART has demonstrated that casual users stay longer within a system when adaptive navigation support is provided. It also provided evidence that direct guidance works best for users with little previous knowledge while adaptive annotation is most helpful for users with some reasonable subject knowledge.

InterBook system (Brusilovsky & Pesin, 1998), a direct descendant of ELM-ART provided the first authoring platform for Web-based adaptive hypermedia. InterBook has refined the ideas of the adaptive electronic textbook and the traffic light metaphor for adaptive navigation support in educational hypermedia (Figure 3). Propagated by ELM-ART and InterBook, this metaphor has later been used in numerous adaptive educational hypermedia systems, including AST (Specht et al., 1997), KBS-HyperBook (Henze & Nejdl, 2001), and SIGUE (Carmona et al., 2002). A study of InterBook has shown that adaptive navigation support encourages non-sequential navigation and helps users who follow the system's guidance to achieve a better level of knowledge.

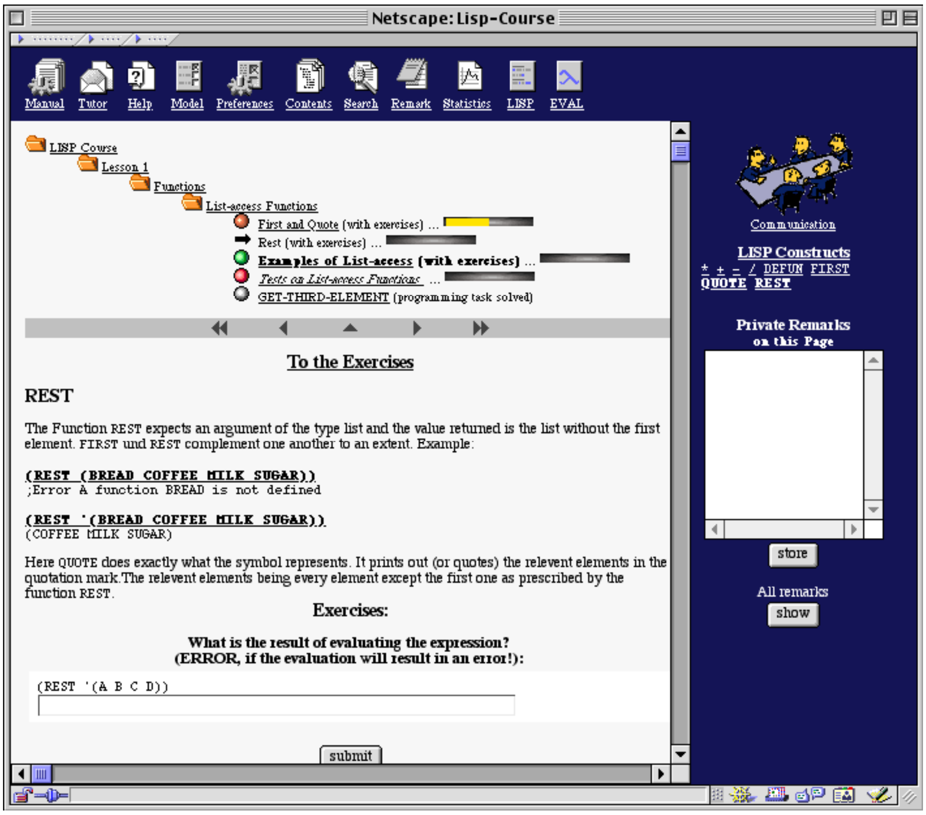


Fig. 2: Adaptive navigation support in ELM-ART. Adaptive annotation is provided in the form of colored bullets following the traffic light metaphor.

ELM-ART and InterBook have also explored a relatively new adaptive navigation support technology known as *link generation*. This technology became very popular in Web hypermedia with its abundance of resources. Unlike classic annotation, sorting or hiding technologies that adapt the presentation of pre-authored links, link generation creates new, non-authored links on a page. There are three known kinds of link generation: discovering new useful links between documents and adding them permanently to the set of existing links; generating links for similarity-based navigation between items; and dynamic recommendation of relevant links. The first two kinds of link generation are typically non-adaptive, though ELM-ART did explore an opportunity to use an episodic student model to generate adaptive links for similarity-based navigation. The third technology is naturally adaptive. It became immensely popular in the field of adaptive Web-based systems through the use of so-called Web recommender systems (Resnick & Varian, 1997). InterBook was among the first systems to have implemented adaptive link generation. It has also demonstrated that link generation can be naturally used in combination with link sorting and annotation.

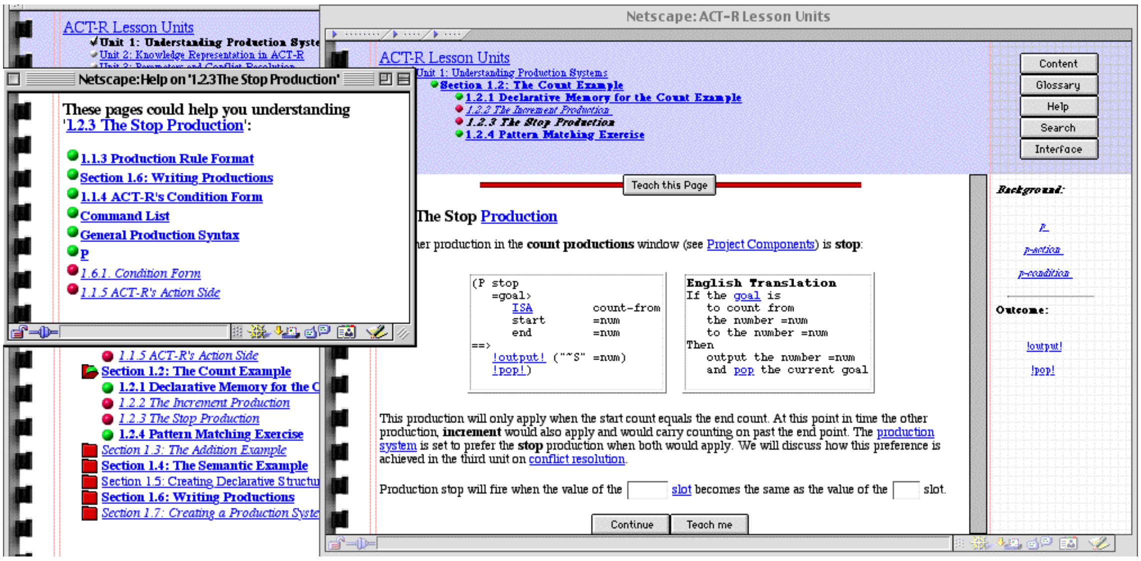


Fig: 3: Adaptive navigation support in InterBook. The system features several kinds of adaptive annotations. In addition, the help recommendation window (left) uses link generation and sorting.

4. From Adaptive Hypermedia to the Adaptive Web

Web-based adaptive hypermedia systems have demonstrated the power of adaptive navigation support in a number of application areas. Yet, they have failed so far to make adaptive navigation support widely available. The problem is that current adaptive navigation support technologies are only applicable within a relatively small set of documents that were structured and enhanced by metadata annotations at design time. Modern AH systems are predominantly *closed corpus* adaptive hypermedia since the *document space* of these adaptive systems is a closed set of information items manually indexed by domain experts. None of the classic adaptive hypermedia systems are applicable in *open corpus* (such as the Web). Closed corpus AH systems demonstrate what is possible to achieve with adaptive hypermedia, but they are impractical for most real world applications. No one is able to invest enough time to structure and index thousands of documents collected from all over the Web so that the result satisfies the requirements of modern adaptive hypermedia systems.

So far, the only adaptive systems that have achieved relative success in working with the open corpus Web are adaptive Web recommendation systems (AWR). Similar to AH, AWR systems support the user in the process of browsing a collection of information resources and use as the source of personalization the observed user activity: link selection, explicit document ratings and various actions indicating implicit interests (Claypool et al., 2001). Unlike AH, AWR employs different kinds of personalization mechanisms that work with open corpus documents: content-based filtering and collaborative filtering (Hanani, Shapira & Shoval, 2001). Content-based filtering relies on word-level document representation and user profiles inherited from information retrieval research and usually employs some machine learning technologies. Collaborative filtering is a social navigation technology (Dieberger et al., 2000) and relies on recorded information about past usage of the same set of documents by multiple users. It applies various profile matching algorithms to match users with similar interests.

There is, however, another important difference between AH and AWR systems that is critical in the context of adaptive navigation support. AH systems attempt to adapt to various "aspects" of the user (goals, knowledge, interests, browsing history) and apply a rich set of adaptive navigation support techniques to express several aspects important to the users at the same time (such as goal relevance, novelty, readiness, etc.). AWR focus their adaptation on one aspect that can be loosely classified as "user interest". The personalization power of AWR is delivered through a less expressive one-dimensional form: a list of recommended links ordered by their perceived interest. Thus, despite their ability to handle open corpus documents, AWR are far from offering Web users the full power of adaptive navigation support as it is offered by classic AH systems.

We think that the current challenge is to develop adaptive navigation support technologies that can work with open corpus documents, i.e., the real Web. The existing technologies that are based on knowledge about documents will be quite useful in the future generation "semantic" Web. However, the best candidates to fuel adaptive navigation support for the current Web are the technologies used in AWR: information retrieval technologies treating a page as a "bag of words" and social navigation technologies rating documents by assembling the "collective wisdom" of their users.

We are investigating the feasibility of using content-based and social technologies for open corpus adaptive navigation support in a more recent project called Knowledge Sea. The first version of the Knowledge Sea system has focused on using content-based IR technologies (Self-Organized Maps) to develop a browseable hyperspace from a set of relatively independent open corpus items and to provide map-based horizontal navigation between open and closed corpus items (Brusilovsky & Rizzo, 2002a). The system was used in a practical context: to provide access to several online tutorials on the C language, as part of a programming course. Knowledge Sea has been evaluated in several user studies (Brusilovsky & Rizzo, 2002b). Students highly rated the system ability to help in selecting relevant open corpus sources, yet most of them have agreed that additional navigation support would be very useful. The Knowledge See II system (Brusilovsky & Chavan, 2003) coupled with AnnotatED social navigation system explored some simple forms of social navigation based on group user modeling and the idea of “footprints” (Wexelblat & Mayes, 1997). It uses the simplest implicit feedback: for each tutorial page it counts how many times it was accessed by a group of users. This amount of traffic is visualized as a color density that students observe during navigation. Each resource is annotated by a blue human-shaped icon on a blue background (right on Figure 4). The deeper the shade of blue the more times the page was accessed. The color of the icon shows the user’s own navigation history, while the color of the background shows the cumulative navigation history of the group the user belongs to (i.e., a class). The color difference between the icon and the background visualizes the discrepancy between user and class navigation patterns. Light figure on a dark background indicates pages that the user accessed less frequently than the average person in his or her group and suggests that these pages deserve attention. A dark figure on a light background indicates a page that the user has accessed more than the group’s average. The same approach is used to annotate horizontal links between pages (i.e., links provided by an author of a particular tutorial. Similarly, the icon/background colors for a knowledge map cell (left on Figure 4) shows total user/class navigation for all pages belonging to the cell.

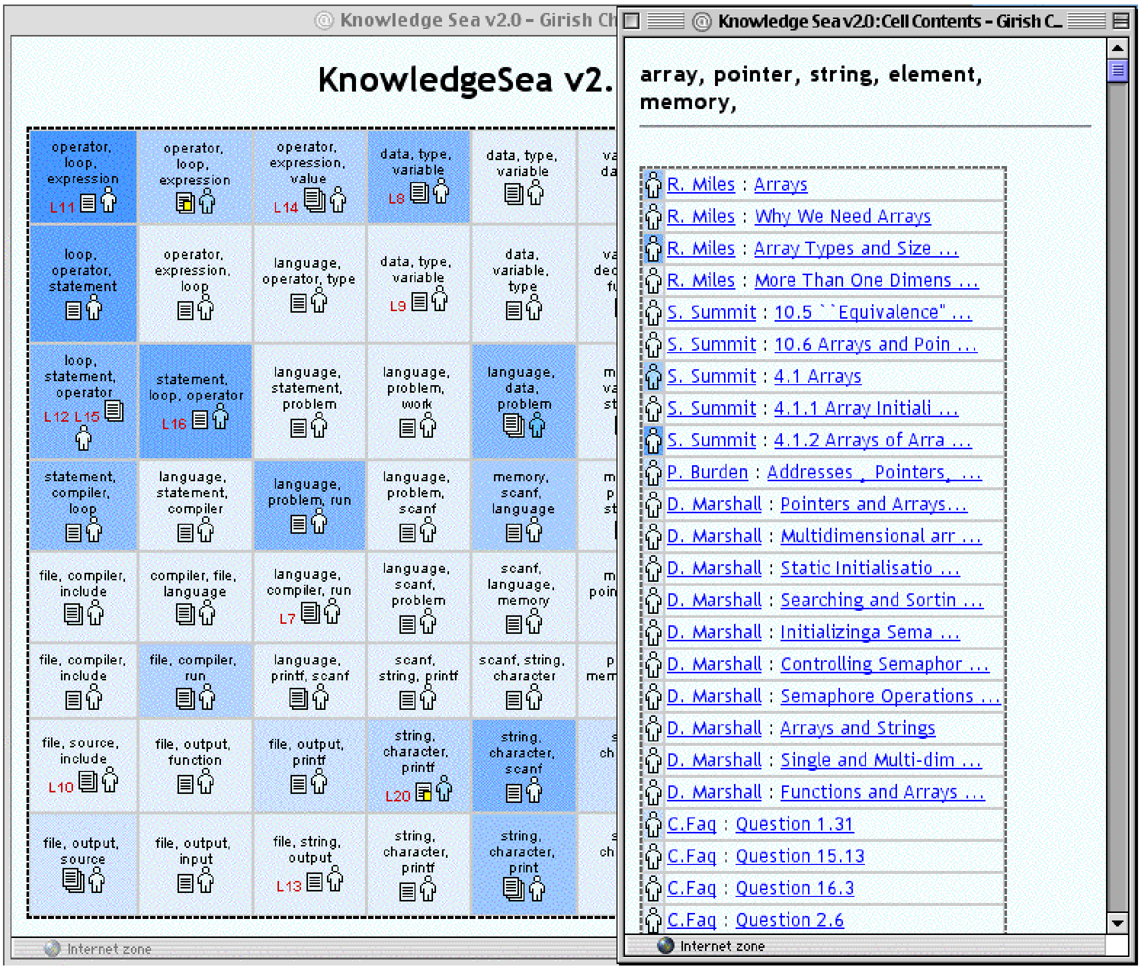


Fig. 4. The interface of Knowledge Sea II systems showing simple social adaptive navigation support.

We have just completed the first classroom formative evaluation of Knowledge Sea II, which brought some interesting results. On one hand, students have appreciated annotation-based SANS: 86% of students considered the visualization of group traffic as useful (in contrast, only 57% considered the visualization of their personal traffic as useful). On the other hand, a number of students realized that simple footprint-style social adaptive navigation support provides insufficient support in context of their needs. In their free-form answers they indicated their interest in seeing some measure of resource quality: ("resources regarded as helpful by classmates", links "rated at a level of importance"), and the relevance of resources to their course needs ("links could be marked as to which lesson they pertain to"). We plan to address these concerns in future work.

5. Adaptive Navigation Support beyond Web-based Hypermedia

Web-based adaptive hypermedia systems have demonstrated their ability to help individual users of hypermedia systems. However, a hyperspace of connected pages - that is the context of existing AH technologies - is no more the only kind of "virtual spaces" that is available for Web users. With advances in delivering complex graphics through the web, virtual reality provides Web users an access to different type of virtual spaces for browsing and exploration. The hyperspace and the 3D virtual environments (VE) are quite different in the nature of their area of applicability, yet there is a striking similarity. Both kinds of cyberspace are targeted for user-driven navigation and exploration. In both kinds of spaces, users can benefit from a personalized support provided by an adaptive intelligent system. We believe that starting from adaptive hypermedia and exploring similarities between hypermedia and 3D virtual environments, it is possible to develop interesting support technologies for 3D virtual environments.

A few known attempts to explore the power of adaptive presentation in the VE context deal with dynamic construction of virtual worlds (Chittaro & Ranon, 2000; Chittaro & Ranon, 2002). Our recent work focuses on adaptive navigation support for VE (Hughes, Brusilovsky & Lewis, 2002). Similar to the case of hypermedia, an adaptive virtual environment can help the user to work more efficiently and avoid common problems such as navigation in the wrong direction, overlooking an important part of the space, and being lost. Our main goal is to develop and evaluate a set of adaptive navigation support techniques for VE by drawing parallels between hypermedia and VE. We are confident that many of the techniques that are employed for adaptive navigation support in hypermedia systems can be extended to 3D visualizations. The paper (Hughes et al., 2002) presents our first attempt to suggest several adaptive navigation support technologies for VE that have clear analogs in hypermedia (i.e., direct guidance, adaptive annotation, etc). Since the original publication, most of the suggested techniques were implemented. Similar techniques have been also implemented by some other teams (Chittaro, Ranon & Ieronutti, 2003). The evaluation data that has been obtained so far delivers some solid evidence in favor of adaptive navigation support in VE. We think that this direction of work will become more important as VE becomes more widely available on the Web and we intend to continue work in this direction.

Fig. 5: Different kinds of adaptive navigation support in Virtual Environments - from direct guidance to annotation. To watch videos, go to www.psychnology.org/article501.htm.

6. Summary

Adaptive navigation support in "virtual spaces" is one of the major research topics of our group. We have explored adaptive navigation support technologies in different contexts from classic adaptive hypermedia to the adaptive Web to virtual environments. We have applied it in different domains from education (Brusilovsky, Eklund & Schwarz, 1998) to avionics performance support (Brusilovsky & Cooper, 2002). We have also investigated the use of different mechanisms to "fuel" the adaptive navigation support - from knowledge-based mechanisms of classic adaptive hypermedia to content-based and social navigation mechanisms of the adaptive Web. Currently, an important focus of our work is to extend the applicability of adaptive navigation support beyond classic adaptive hypermedia. Our work on adaptive navigation support for open corpus Web hypermedia and virtual environments represents our most recent efforts in this direction. By extending the borders of adaptive navigation support we hope to better understand this technology and to allow more users to benefit from it.

7. Acknowledgements

The author thanks Stephen Hughes, the developer of adaptive navigation support techniques for VE for his help with preparing illustrative material for this paper.

8. References

Böcker, H. D., Hohl, H., and Schwab, T. (1990) °pAdapter - Individualizing Hypertext. In: D. Diaper (ed.) *INTERACT'90*. (Proceedings of IFIP TC13 Third International

Conference on Human-Computer Interaction, Cambridge, 27-31 August, 1990) Amsterdam: North-Holland, pp. 931-936.

Brusilovsky, P. (1992) A framework for intelligent knowledge sequencing and task sequencing. In: C. Frasson, G. Gauthier and G. McCalla (eds.) *Intelligent Tutoring Systems*. (Proceedings of Second International Conference on Intelligent Tutoring Systems, ITS'92, Montreal, June 10-12, 1992) Berlin: Springer-Verlag, pp. 499-506.

Brusilovsky, P. (1996) Methods and techniques of adaptive hypermedia. In P. Brusilovsky and J. Vassileva (eds.), *User Modeling and User-Adapted Interaction* 6 (2-3), Special Issue on Adaptive Hypertext and Hypermedia, 87-129.

Brusilovsky, P. (1997) Efficient techniques for adaptive hypermedia. In: C. Nicholas and J. Mayfield (eds.): *Intelligent hypertext: Advanced techniques for the World Wide Web*. Lecture Notes in Computer Science, Vol. 1326, Berlin: Springer-Verlag, pp. 12-30.

Brusilovsky, P. (2001) Adaptive hypermedia. *User Modeling and User Adapted Interaction* 11 (1/2), 87-110, also available at http://www.wkap.nl/oasis.htm/270983.

Brusilovsky, P. and Chavan, G. (2003) Knowledge Sea II, University of Pittsburgh, Pittsburgh. http://www.pitt.sis.edu/~taler/KnowledgeSea.html (Accessed December 15, 2003)

Brusilovsky, P. and Cooper, D. W. (2002) Domain, Task, and User Models for an Adaptive Hypermedia Performance Support System. In: Y. Gil and D. B. Leake (eds.) Proceedings of 2002 International Conference on Intelligent User Interfaces, San Francisco, CA, January 13-16, 2002, ACM Press, pp. 23-30.

Brusilovsky, P., Eklund, J., and Schwarz, E. (1998) Web-based education for all: A tool for developing adaptive courseware. *Computer Networks and ISDN Systems* (Proceedings of Seventh International World Wide Web Conference, 14-18 April 1998) 30 (1-7), 291-300.

Brusilovsky, P. and Maybury, M. T. (2002) From adaptive hypermedia to adaptive Web. In P. Brusilovsky and M. T. Maybury (eds.), *Communications of the ACM* 45 (5), Special Issue on the Adaptive Web, 31-33.

Brusilovsky, P. and Pesin, L. (1994) ISIS-Tutor: An adaptive hypertext learning environment. In: H. Ueno and V. Stefanuk (eds.) Proceedings of JCKBSE'94, Japanese-CIS Symposium on knowledge-based software engineering, Pereslavl-Zalesski, Russia, May 10-13, 1994, EIC, pp. 83-87.

Brusilovsky, P. and Pesin, L. (1998) Adaptive navigation support in educational hypermedia: An evaluation of the ISIS-Tutor. *Journal of Computing and Information Technology* 6 (1), 27-38.

Brusilovsky, P., Pesin, L., and Zyryanov, M. (1993) Towards an adaptive hypermedia component for an intelligent learning environment. In: L. J. Bass, J. Gornostaev and C. Unger (eds.) *Human-Computer Interaction*. Lecture Notes in Computer Science, Vol. 753, (Proceedings of 3rd International Conference on Human-Computer Interaction, EWHCI'93, Moscow, Russia, August 3-7, 1993) Berlin: Springer-Verlag, pp. 348-358.

Brusilovsky, P. and Rizzo, R. (2002a) Map-based horizontal navigation in educational hypertext. *Journal of Digital Information* 3 (1), also available at http://jodi.ecs.soton.ac.uk/Articles/v03/i01/Brusilovsky/.

Brusilovsky, P. and Rizzo, R. (2002b) Using maps and landmarks for navigation between closed and open corpus hyperspace in Web-based education. *The New Review of Hypermedia and Multimedia* 9, 59-82.

Brusilovsky, P., Schwarz, E., and Weber, G. (1996) ELM-ART: An intelligent tutoring system on World Wide Web. In: C. Frasson, G. Gauthier and A. Lesgold (eds.) *Intelligent Tutoring Systems*. Lecture Notes in Computer Science, Vol. 1086, (Proceedings of Third International Conference on Intelligent Tutoring Systems, ITS-96, Montreal, June 12-14, 1996) Berlin: Springer Verlag, pp. 261-269, also available at http://www.contrib.andrew.cmu.edu/~plb/ITS96.html.

Carmona, C., Bueno, D., EduardoGuzman, and Conejo, R. (2002) SIGUE: Making Web Courses Adaptive. In: P. De Bra, P. Brusilovsky and R. Conejo (eds.) Proceedings of Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002) Proceedings, Málaga, Spain, May 29-31, 2002, pp. 376-379.

Chittaro, L. and Ranon, R. (2000) Adding adaptive features to virtual reality interfaces for ecommerce. In: P. Brusilovsky, O. Stock and C. Strapparava (eds.) *Adaptive Hypermedia and Adaptive Web-Based Systems*. Lecture Notes in Computer Science, (Proceedings of Adaptive Hypermedia and Adaptive Web-based Systems, AH2000, Trento, Italy, August 28-30, 2000) Berlin: Springer-Verlag, pp. 86-91.

Chittaro, L. and Ranon, R. (2002) Dynamic generation of personalized VRML content: a general approach and its application to 3D E-commerce. In: Proceedings of 7th International Conference on 3D Web Technology, Web3D'2002, , ACM Press, pp. 145-154.

Chittaro, L., Ranon, R., and Ieronutti, L. (2003) Navigating 3D worlds by following virtual guides: helping both users and developers of 3D Web sites. In: L. Ardissono and A. Goy (eds.) Proceedings of HCIITALY'2003, Turin, Italy, November 4-5, 2003, University of Turin, pp. 42-47.

Claypool, M., Le, P., Wased, M., and Brown, D. (2001) Implicit interest indicators. In: Proceedings of 6th International Conference on Intelligent User Interfaces, Santa Fe, NM, January 14-17, 2002, ACM Press, pp. 33-40.

De Bra, P. and Calvi, L. (1998) AHA! An open Adaptive Hypermedia Architecture. In P. Brusilovsky and M. Milosavljevic (eds.), *The New Review of Hypermedia and Multimedia* 4, Special Issue on Adaptivity and user modeling in hypermedia systems, 115-139.

de La Passardiere, B. and Dufresne, A. (1992) Adaptive navigational tools for educational hypermedia. In: I. Tomek (ed.) *Computer Assisted Learning*. (Proceedings of ICCAL'92, 4-th International Conference on Computers and Learning, Wolfville, Canada, June 17-20, 1992) Berlin: Springer-Verlag, pp. 555-567.

Dieberger, A., Dourish, P., Höök, K., Resnick, P., and Wexelblat, A. (2000) Social Navigation: Techniques for Building More Usable Systems. *interactions* 7 (6), 36-45.

Hammond, N. (1989) Hypermedia and learning: Who guides whom? In: H. Maurer (ed.) *Computer Assisted Learning*. Lecture Notes in Computer Science, Vol. 360, (Proceedings of 2-nd International Conference on Computer Assisted Learning, ICCAL'89, Dallas, TX, May 9-11, 1989) Berlin: Springer-Verlag, pp. 167-181.

Hanani, U., Shapira, B., and Shoval, P. (2001) Information filtering: Overview of issues, research and systems. *User Modeling and User Adapted Interaction* 11 (3), 203-259.

Henze, N. and Nejdl, W. (2001) Adaptation in open corpus hypermedia. In P. Brusilovsky and C. Peylo (eds.), *International Journal of Artificial Intelligence in Education* 12 (4), Special Issue on Special Issue on Adaptive and Intelligent Web-based Educational Systems, 325-350, also available at http://cbl.leeds.ac.uk/ijaied/abstracts/Vol_12/henze.html.

Hohl, H., Böcker, H.-D., and Gunzenhäuser, R. (1996) Hypadapter: An adaptive hypertext system for exploratory learning and programming. In P. Brusilovsky and J. Vassileva (eds.), *User Modeling and User-Adapted Interaction* 6 (2-3), Special Issue on Adaptive Hypertext and Hypermedia, 131-156.

Hughes, S., Brusilovsky, P., and Lewis, M. (2002) Adaptive navigation support in 3D e-commerce activities. In: F. Ricci and B. Smyth (eds.) Proceedings of Workshop on Recommendation and Personalization in eCommerce at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002) Proceedings, Málaga, Spain, May 28, 2002, pp. 132-139, also available at http://ectrl.itc.it/home/rpec/RPEC-Papers/15-hughes.pdf.

Kaplan, C., Fenwick, J., and Chen, J. (1993) Adaptive hypertext navigation based on user goals and context. *User Modeling and User-Adapted Interaction* 3 (3), Special Issue on User models & user-adapt. interact., 193-220.

Pérez, T., Gutiérrez, J., and Lopistéguy, P. (1995) An adaptive hypermedia system. In: J. Greer (ed.) Proceedings of AI-ED'95, 7th World Conference on Artificial Intelligence in Education, Washington, DC, 16-19 August 1995, AACE, pp. 351-358.

Resnick, P. and Varian, H. R. (1997) Recommender systems. *Communications of the ACM* 40 (3), 56-58.

Specht, M., Weber, G., Heitmeyer, S., and Schöch, V. (1997) AST: Adaptive WWW-Courseware for Statistics. In: P. Brusilovsky, J. Fink and J. Kay (eds.) Proceedings of Workshop "Adaptive Systems and User Modeling on the World Wide Web" at 6th International Conference on User Modeling, UM97, Chia Laguna, Sardinia, Italy, June 2, 1997, Carnegie Mellon Online, pp. 91-95, also available at http://www.contrib.andrew.cmu.edu/~plb/UM97_workshop/Specht.html.

Weber, G. and Brusilovsky, P. (2001) ELM-ART: An adaptive versatile system for Web-based instruction. In P. Brusilovsky and C. Peylo (eds.), *International Journal of Artificial Intelligence in Education* 12 (4), Special Issue on Adaptive and Intelligent Web-based Educational Systems, 351-384, also available at http://cbl.leeds.ac.uk/ijaied/abstracts/Vol_12/weber.html.

Wexelblat, A. and Mayes, P. (1997) Footprints: History rich Web browsing. In: Proceedings of Conference on Computer-Assisted Information Retrieval, , pp. 75-84.

Zeiliger, R. (1993) Adaptive testing: contribution of the SHIVA model. In: D. Leclercq and J. Bruno (eds.): *Item banking: Interactive testing and self-assessment*. NATO ASI Serie F, Vol. 112, Berlin: Springer-Verlag, pp. 54-65.

PsychNology Journal, 2004
Volume 2, Number 1, 24 - 42

Navigating 3D Virtual Environments by Following Embodied Agents: a Proposal and its Informal Evaluation on a Virtual Museum Application

Luca Chittaro♣, Lucio Ieronutti, Roberto Ranon

HCI Lab
University of Udine, Italy

ABSTRACT

Many 3D virtual environments (e.g., 3D Web sites) do not offer sufficient assistance to (especially novice) users in navigating the virtual environment, find objects/places of interests, and learn how to interact with them. This paper proposes the adoption of guided tours of virtual environments as an effective user aid and describes a novel tool that provides automatic code generation for adding such guided tours to 3D virtual environments developed using the VRML language. In the second part of the paper, we informally evaluate the proposed approach on a real-world application concerning a 3D computer science museum (a complement to a real-world exhibition focusing on the history of computer technology).

Keywords: *3D Virtual Environments, Navigation Aids, Virtual Museums, Embodied Agents.*

Received 16 January 2004; received in revised form 4 April 2004; accepted 2 April 2004.

1. Introduction

A well-known factor limiting the diffusion and popularity of 3D interfaces (e.g., 3D Web sites on the Internet) is their scarce usability (especially from the point of view of novice users). While developers of traditional 2D interfaces (e.g., traditional 2D Web sites) are aware that usability is one of the key issues for the success of their applications and rely on guidelines and tools that support them in this direction, 3D content creators have generally neither well-established 3D-specific guidelines nor specific tools available to help them.

In our research, we are concentrating on finding ways to increase the usability of 3D interfaces. In this paper, we consider a specific usability problem that affects many 3D

♣ Corresponding Author:
Luca Chittaro
Dept. Of Math and Computer Science
Via delle Scienze, 206
33100 Udine, Italy
email: chittaro@dimi.uniud.it

virtual environments, i.e. insufficient user assistance during exploration, and propose an approach aimed at helping the 3D content creator to face the problem with little effort.

The considered solution is based on exploiting an embodied agent, making it able to lead the user on a guided tour of the 3D virtual environment. This solution is at the same time a navigation aid (since it helps users in finding places of interest) and an information aid (since the agent is also able to provide information about the encountered places and objects). The introduction of an embodied agent can also have the additional advantage of making the 3D virtual environment more lively and attractive for the user.

Unfortunately, developing guided tours led by embodied agents is not an easy task for the 3D content creator, since it currently has to be done partly by hand (e.g., coding a suitable path for the virtual agent avoiding obstacles). Moreover, the code written for one 3D virtual environment can be very limitedly reused for other ones. The aim of this paper is to aid the 3D content creator by proposing an automatic approach to solve the considered problem. The current implementation of the approach is targeted at 3D virtual environments developed using the VRML language (VRML, 1997), a ISO standard language to develop 3D content for the Internet. The detailed description on how the approach has been implemented can be found in (Chittaro et al., 2003), while this paper focuses on the conceptual features of the approach and its informal evaluation.

The informal evaluation of the approach has been carried out on an application concerning a 3D virtual museum we recently developed as a complement to a real-world exhibition focusing on the history of computer technology. The 3D virtual museum application allows one to visit a virtual reconstruction of a computing centre of the '70s that shows the corresponding equipment of the real-world exhibition in its original environment of operation and provides information about its features and functioning. To make the visiting experience more engaging and effective, we exploited the tool proposed in this paper to build a human-like agent that acts as a virtual guide, being able to lead the user on a guided tour explaining some/all the objects in the museum in a logical order.

This paper is structured as follows. First, in Section 2, we advocate the exploitation of virtual tours with embodied agents as guides to effectively aid users in 3D virtual environments, and compare it with other proposed navigation aids (e.g. maps, viewpoint tours). In Section 3, we illustrate our approach to build guided tours with embodied agents, and show how a 3D content creator can use it as a stand-alone tool for automatic code generation. Section 4 describes the 3D Computer Science museum and illustrates the capabilities of its virtual guide. In Section 5, we present the results of an informal evaluation of the museum virtual guide on users, and draw some interesting lessons on the exploitation of embodied agents in 3D museums and exhibitions.

2. Motivations

Many 3D virtual environments, whether representing existing places (e.g., virtual cities) or imaginary ones, typically leave the user alone and partially or totally unassisted in navigating the environment, discover points of interest, and interact with objects. Although in games this can be a desirable situation, it is definitely not in applications devoted to other purposes (such as virtual tourism, training, museums, e-commerce,...). Leaving the user unassisted can lead to a number of usability problems, ranging from navigation issues (e.g., wayfinding) to difficulties in figuring out which operations can be performed on the objects in the virtual environment. In the following, we analyze these issues in detail.

2.1. Navigation Issues

The lack of proper navigation support causes the user to suffer from well-known navigation problems (e.g., disorientation). As a result, visitors of a virtual environment can: (i) become rapidly frustrated and abandon the visit; (ii) miss interesting parts of the environment (especially in large ones), and (iii) complete the visit with the feeling of not having adequately explored the environment.

This is particularly true for novice users of a virtual environment, who should be helped as much as possible during navigation by offering them proper navigation aids. To this purpose, a first category of solutions can be derived from the design of real-world environments. For example, in the design of buildings, architects aim at reducing wayfinding problems for the people exploring the building by increasing visual access (i.e. the number of parts of the environment which can be seen by a person from her position in space) or including navigational cues (e.g., room numbers, names of buildings, landmarks). Landmarks are distinctive environmental features (e.g., a statue, a river, a town square, …) functioning as reference points (Vinson, 1999). However, this kind of solutions can be limitedly applied to 3D virtual environments that are virtual reconstructions of existing places, where the architecture of the environment cannot be modified.

A second category of solutions (that can be possibly combined with the above mentioned ones) aims at providing the user with electronic navigation aids to augment her capabilities to explore and learn. A well known example in this category are the various types of electronic maps of the environment that help users orient themselves (see, e.g., (Darken and Sibert, 1996)). However, electronic maps adopt a third-person perspective, which can require a considerable cognitive mapping effort to be correctly interpreted (e.g., consider the typical real-world situation where someone is trying to find her way in a city by using a map and has to translate the exocentric view of the map into her egocentric view). This claim is supported by psychological studies, e.g. Thorndyke and Hayes-Roth (1982) compared spatial judgment abilities of subjects who learned an environment from personal exploration or from a map, highlighting the difficulty of changing perspective (e.g., subjects who acquired knowledge from the exocentric map perspective were most error prone in tasks that required to translate their knowledge into a response within the environment). Other classes of proposed navigation aids include:

- 3D maps, such as Worlds in Miniature (WIM) (Stoakley et al., 1995). A WIM is a three-dimensional small scale version of the 3D virtual environment, standing in front of the user, as if it were in his virtual hand. The user can directly manipulate both the WIM and the environment (changing something in one of the two directly affects the other and vice versa);
- constrained navigation approaches, that restrict user's freedom of movement. Placing constraints on user's motion allows for a simplification of the effort needed to navigate the 3D virtual environment. For example, Galyean (1995) has proposed a method that guides the user’s continuous and direct input within both space and time using a river analogy, while Hanson and Wernert (1999) present an approach where a user's navigation can be constrained to the movements of another user or of an automated guide.

In the following, we focus on a solution belonging to the navigation aids category.

2.2. Getting Information about the Virtual Environment

Another difficulty that can be experienced by the visitor of a virtual environment is to determine what she can do with objects and how to get information about them. Instructions to the user are typically provided by introductory pages, which describe the contents of the virtual environment, and how to interact with objects. Many users typically skip these introductory pages and later fail to explore parts of the virtual environment because they are not aware of all the possible actions.

Providing information about objects can be crucial, for example, in virtual museums, virtual training and e-commerce sites, where getting additional information is a fundamental part of the user experience. For example, in the case of 3D Web sites, some researchers face this problem by providing the information during the visit in a separate HTML frame. However, this solution has the disadvantage of decreasing user immersion.

2.3. A Possible Solution: Using Embodied Agents as Guides

One of the solutions that can take into account all the previous concerns is to offer tours of the 3D virtual environment. Moreover, from the 3D content creator perspective, the possibility of offering tours allows one to naturally suggest preferable ways to visit the virtual environment. For example, this can be important in virtual exhibitions, museums, and cities.

Simple forms of (unguided) tours are already offered by some 3D virtual environments by providing a set of viewpoints through which the user has to cycle to visit the environment in a certain order. However, while viewpoints can be useful for quickly navigating the environment, they are not easy to use for learning navigational knowledge, such as paths, relating the different parts of the environment (if teleporting is used, they also break the continuity of the experience, further contributing to user disorientation).

The approach we adopt in this paper is to provide the user with guided tours based on an embodied (more specifically, a humanoid) agent that acts as a *virtual guide*, i.e. it is able both to lead the user to the required places and to provide information through a semi-transparent *On Screen Display* (*OSD*) available inside the 3D virtual environment. In the following, we discuss in detail how this approach addresses the previously illustrated concerns, and the benefits of using embodied agents as virtual guides.

First, virtual guides are navigation aids, leading users around and preventing them from being lost: the user has simply to follow the guide to visit the virtual environment. As pointed out by Rickel and Lewis Johnson (2000), showing the user where relevant objects are and how to get to them is likely to be more effective than trying to tell users where objects are located. For example, informal experiments with users in a virtual city showed that the environment was explored to a greater level and by an higher number of users as a result of tour guides explaining how to get the most out of the system (Ennis and Mayer, 2001). Moreover, the expert user has still the possibility to ignore the guide and follow her own tour, but it is very likely that novice users will instead appreciate and use the aid. While a virtual guide does not directly help the user properly controlling its movements to avoid some typical 3D navigation problems (e.g., bumping into obstacles), a properly chosen tour can lower the probability that the user finds himself in positions where it can be difficult to find a way (e.g. corners).

Second, virtual guides can be also employed as a natural way to provide users with additional information (e.g. usage of a particular object) during the visit, instead of providing this information into separate pages. For example, a virtual museum guide could give an introduction at the beginning of the visit and later present specific items on display, possibly explaining how to interact with them. With respect to other forms of user guidance, an embodied agent can draw user’s attention with the most common and natural methods, such as gaze and pointing gestures (Rickel and Lewis Johnson, 2000). Moreover, the guide can also use its body orientation as a cue to the suggested attentional focus (Rickel and Lewis Johnson, 2000).

Third, introducing an embodied agent in the virtual environment can contribute to alleviate an additional problem, i.e. most virtual environments where the user is left completely alone look like dead places (like just after the builders have left the building), that are less attractive to the user. Embodied agents can make instead the virtual place more lively, attractive, and less intimidating. Results of empirical studies show that embodied agents may have a strong motivational impact: users tend to experience presentations given by embodied agents as lively and engaging (Lester et al. 1997; van Mulken et al. 1998).

Finally, we concentrated our attention on guides that travel around the virtual environment by simply walking (i.e., they do not fly, crawl, ...). Indeed, it would be very difficult for the user to follow paths that require movements other than walking, e.g., free flying in a 3D space is known to be very difficult for novice users.

3. Deriving Virtual Tours for 3D Virtual Environments with Virtual Guides

Achieving the previously illustrated goals requires to deal with a number of different problems, ranging from the derivation of suitable paths for the virtual guide, to choosing how to present objects/places of interest to the user.

The tool we present, called *VRML Tour Creator*, aims at providing fully automatic code generation, allowing the content creator to concentrate only on high-level aspects (e.g., which objects/places of the virtual environment need to be presented, what could be the text for their presentation,...). More specifically, the inputs of the VRML Tour Creator are the VRML file describing the virtual environment in which the guided tour needs to be inserted, the 3D model of the embodied agent which will act as a virtual guide, and the tour data, i.e. the descriptions and the locations of the objects/places to be presented. The tool then outputs a new version of the VRML virtual environment that includes the generated code for the guided tour.

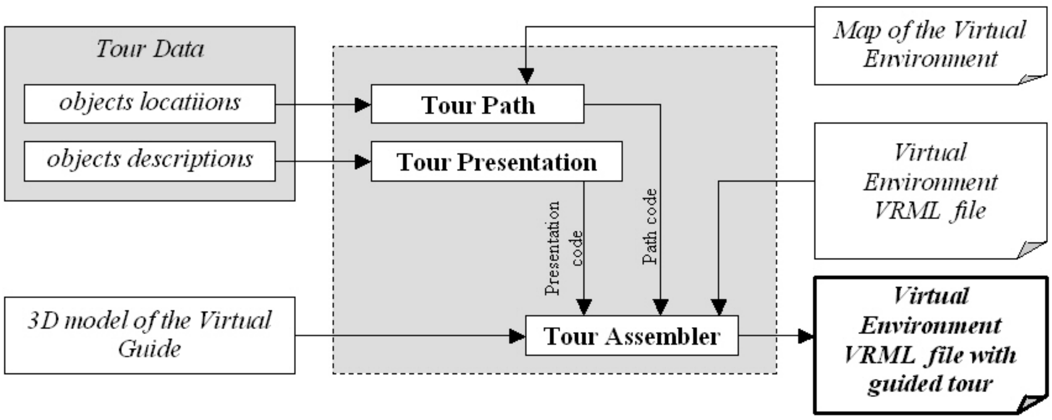


Fig. 1: Architecture of the VRML Tour Creator

The general architecture of the VRML Tour Creator (which has been implemented in Java) is depicted in Figure 1. It is mainly composed by three modules: a *Tour Path* module, a *Tour Presentation* module and a *Tour Assembler* module.

The function of the Tour Path module is to derive an appropriate sequence of positions that will be used to move the guide from each object/place to the subsequent one in the desired tour.

The approach we follow to deal with the problem of deriving a suitable set of positions for the guided tour is based on using a path planning algorithm that takes as input a map of the VRML virtual environment (which is automatically generated using the approach described in (Ieronutti et al., 2004)). In particular, we adopted a modified version of a well-known grid-based path planning approach (Latombe, 1991) from robotics, which provides a good compromise between simplicity and generality, and guarantees that the virtual guide, following the calculated path, will not collide with other objects in the virtual environment (more details can be found in (Chittaro et al., 2003)).

The Tour Presentation module derives the VRML code for the presentation of each object/place in the tour. Each presentation is displayed in the semi-transparent OSD that appears at presentation time and/or is narrated by the embodied agent using a text-to-speech engine.

The Tour Assembler module uses the outputs derived by the two previously described modules together with other inputs to assemble the code for the guided tour and include it in the virtual environment.

4. Case Study: a 3D Virtual Computer Science Museum

3D interfaces are increasingly used in the museum domain, both as a complement and extension of existing real-world museums (e.g., Virtual Reality theatre rooms, such as the CAVE system (Cruz-Neira et al., 1993)) and as a means to build entirely virtual museums (e.g., to be visited from the Web or to be taken on tour). For example, Barbieri and Paolini (2001) developed a multi-user virtual environment that shows Leonardo's works in an application for the Italian National Science Museum, while (Chiu et al., 2000) developed a 3D virtual museum, called Virtual Architecture Museum, focused on modernist asian architecture.

Fig. 2: Some units of a real Univac Sperry 90/30 computer.

Fig. 3. Top view of the virtual data processing centre.

With respect to a 2D solution (e.g., a collection of HTML pages), a 3D virtual museum allows one not only to display the museum items, but also to convey their "cultural setting" (since it constitutes an important part of the display) by inserting them in a proper environment (Hendricks et al., 2003).

Recently, we developed a 3D Computer Science museum based on a VRML virtual environment representing a data processing center of the 70's, reproducing hardware from the Univac/Sperry 90/30 line (shown in Figure 2). In the virtual environment, visitors can either freely navigate or follow guided tours, observe the different devices and people at work, and learn the features and functioning of devices. The main pedagogical goals for our virtual museum are concerned with pointing out the several differences existing between data processing centers in the '70s and current computer machines, e.g. the mainframe – terminals architecture and the interaction based on text video terminals or (more often) punch cards. The virtual data processing centre is divided into two main rooms (as shown in Figure 3):

- a computer room, containing the main system, in which data processing is performed under the control of a technical staff (i.e., experts in computer science or in electronic engineering);

- a terminal room, containing punch card units and video terminals, in which activities that are mainly preparatory to real data processing are carried out by other people, usually less expert in computer science, but more competent in the specific problems to solve.

Museum visitors can click on any device, and see and/or hear a description of its features and functioning.

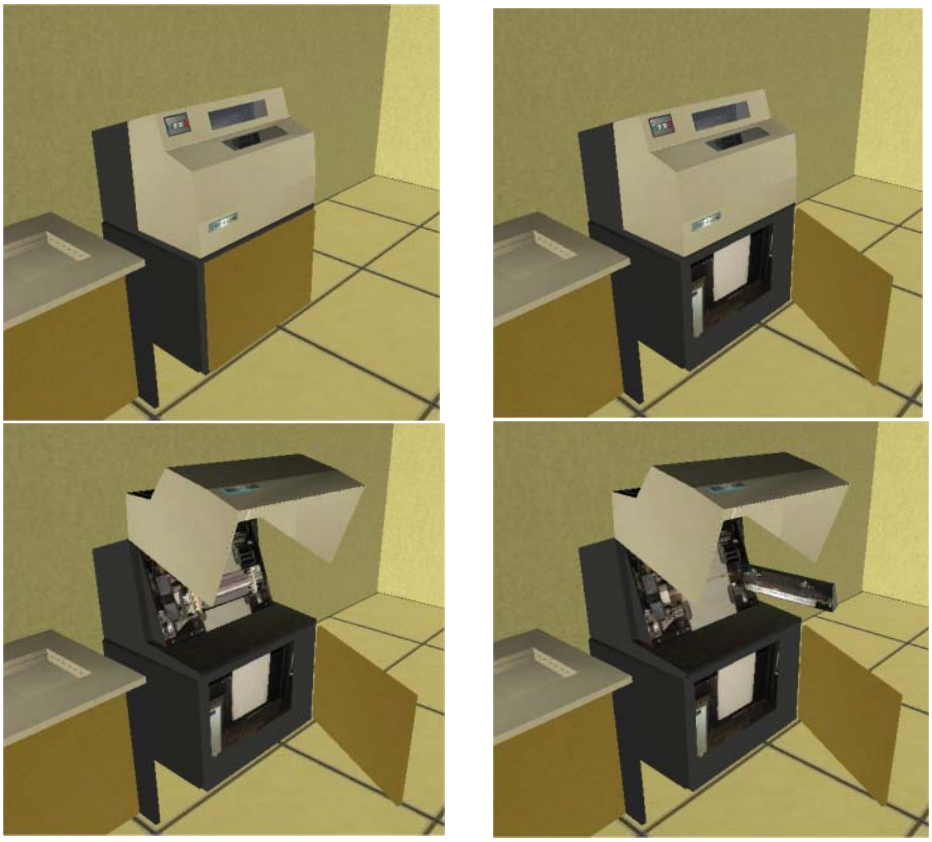


Fig. 4: An interactive component: the main printer.

To improve learning effectiveness, most devices are interactive and some are animated. For example, the user can open cabinet units to examine their internal details and working (as with the main printer illustrated in Figure 4).

To increase the realism of the user experience, we added the necessary furniture and included typical professional figures at work. Moreover, the audio channel is used to add typical noise and sounds of objects and human actions (for instance, printers, terminal operator typing, etc). Noise and sounds augment the immersiveness of the experience of the visitor and also help focusing the attention during the visit.

4.1. A Guided Tour of the Computer Science Museum

Visitors of the museum can either freely navigate or follow a guided tour lead by the embodied agent (shown in Figure 5). As the user enters the 3D Computer Science Museum, the embodied agent performs an introduction to the subject, and invites the user to follow it on a tour, which encompasses some (or all) of the museum items. Each tour is composed by a sequence of presentations (each one referring to one museum item). For each presentation, the guide: (i) walks from its actual position towards the item that has to be presented (showing the user the way to reach it); (ii) presents the item (see Figure 5); (iii) waits until the user clicks on the guide itself, and then proceeds to the next item in the tour. While the presentation text is displayed and/or spoken, the guide mimics a talking person (see Figure 6). Additionally, when presenting an object, the guide periodically switches its attention between the user and the object to both give the user the feeling that the guide is addressing her, and to ensure that the attention towards the required object is clearly directed. Visitors are not forced to follow the guide; they can autonomously visit the environment and possibly come back later and resume the guided tour by clicking again on the guide.

5. Informal Evaluation on Users

The aim of the evaluation we carried out on the 3D Computer Science Museum was twofold: first, we wanted to assess whether the virtual guide was appreciated (and exploited) by users; second, we wanted to identify strengths and weaknesses of the guided tour approach, and get feedback about how to improve the system.

Subjects

A total of 20 subjects were involved in the evaluation. The subject population was representative of the diversity of visitors of the real exhibition. Age ranged from 20 to 55, averaging at 30. The population was almost equally split between males (11) and females (9). Occupation of the subjects ranged from clerk to high school teacher, engineer, and student. With respect to familiarity with 3D environments, 30% of the population was familiar with 3D environments (mostly videogames), 40% of the population had visited only one or two 3D environments, and the remaining 30% had never visited a 3D environment.

Fig. 5. The virtual guide presenting a printer.

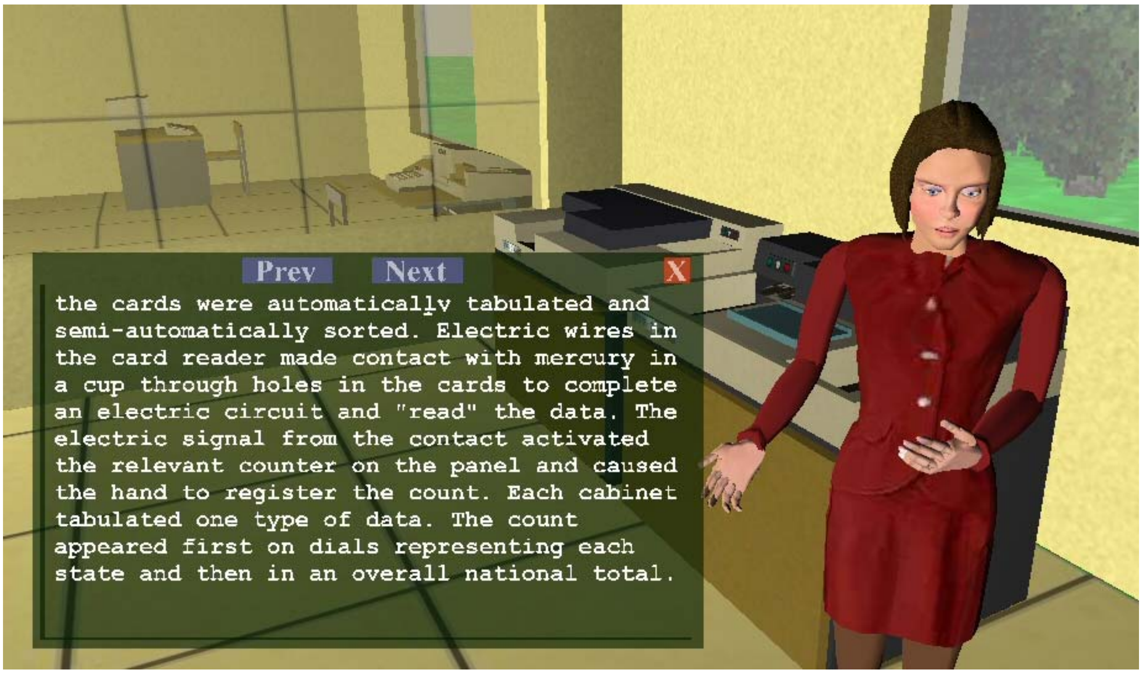


Fig. 6: The required presentation text is read by a text-to-speech engine and/or displayed on a semi-transparent On-Screen Display. The virtual guide mimics a talking person.

Evaluation Setting and User Task

Interaction with the virtual environment took place through keyboard and mouse. Users could choose between mouse and the four arrow keys on the keyboard for movement inside the 3D Computer Science museum, while only the mouse could be used to point at and manipulate objects. The hardware used for the experiment was a standard 19 inches Trinitron monitor and a 2.4 GHz Pentium IV PC equipped with a Nvidia GeForce4 Ti4600 graphics board. The full screen was devoted to present the view of the visited environment.

Prior to the virtual visit, subjects filled out a brief demographic questionnaire. Next, they were instructed about how to interact with the application via keyboard and mouse, using a simple training environment (two connected rooms with objects that can be clicked and moved).

After completing the training phase, the virtual visit started. Subjects were asked to freely wander around the museum and get information about the displayed objects of interest. After entering the museum, they were free to choose whether to follow the embodied agent on a guided tour, which gave a general introduction to the subject and was composed by 9 presentations of museum items, or visit the museum freely. At any time, visitors had the option of abandoning the guided tour (by simply not clicking on the guide after a presentation) or resuming a previously abandoned tour. We explicitly told subjects that following the guide was optional, and that they could obtain the same information by freely exploring the museum and clicking on the museum items.

All users' interactions with the application were logged. In particular, we recorded: (i) each user's path in the virtual environment; (ii) each interaction with the virtual guide (i.e., each click on the virtual guide to continue the tour with the next museum item); (iii) each interaction with museum items (e.g. request of information, opening of cabinets).

After the virtual visit, the subjects filled out a second questionnaire, where they were asked to rate the ease of use of the system and their impressions of the virtual guide. Finally, users were interviewed (using a semi-structured interview protocol) to determine strengths and weaknesses and collect suggestions.

Results

From the point of view of guide usage, 19 out of 20 users chose to exploit the virtual guide (i.e., they requested at least one presentation to the guide). The average number of requests of presentation per user was 6 (out of 9 possible presentations), indicating a considerable exploitation of the guide functionality. It is interesting to note that users with no experience of 3D environments requested more presentations, while users with good familiarity with 3D environments requested less presentations (see

Figure 7). With respect to guide usage, there were no differences related with gender, age, and duration of visit.

Figure 8 visualizes the users' flows on the virtual museum map. Black areas represent the museum building, items and furniture; the white/grey color identifies the areas visited by users (the darker the shade of grey, the longer the time users spent in the area; the white color represents areas where the user spent less time); areas painted with diagonal stripes indicates the positions over which no user stood. Dots represent locations where the guide presented a specific item (in the order given by the numbers in the figure), while black thin lines highlight the guide path along the tour. It is interesting to note that some presentation locations (i.e., 1, 2, 3, 4, 6, 8) correspond to or are very close to highly visited areas, while this is not true for other presentation locations (i.e., 5, 7, 9), which are in areas not easily accessible by users. This indicates that users tend to avoid places where it could not be easy to find a way out (e.g., corners), and highlights the usefulness of the guide in presenting items that would go probably unnoticed because their position is difficult to access.

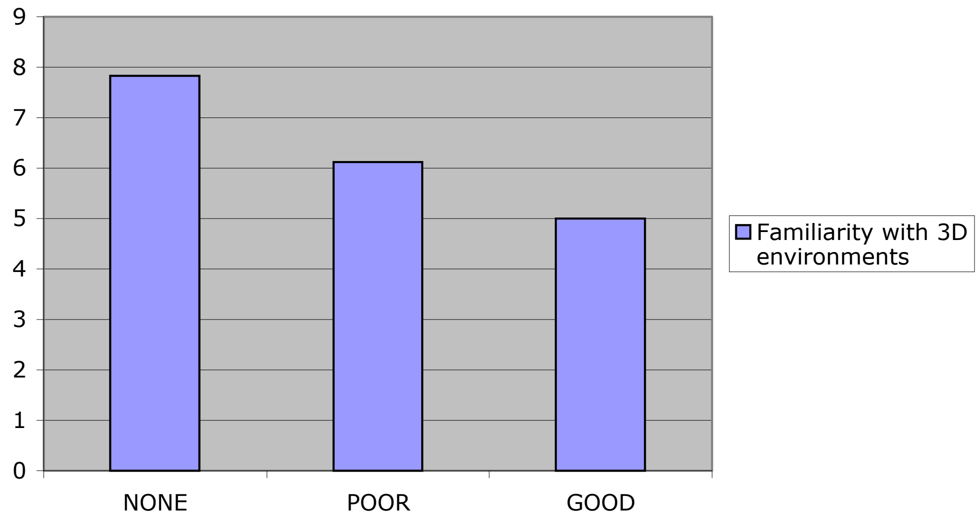


Fig. 7: Average number of presentations requested to the guide by users with none, poor and good experience with 3D navigation.

The users' answers, collected in the questionnaire, are summarized in Table 1. Users were asked to rate the easiness and appeal of the interaction with the system, and the usefulness and walking speed of the guide. While most users found the interaction easy, and rated the virtual guide as useful, reactions to the appeal of the system and the speed of the virtual guide were mixed.

During the final interview, we focused on problematic aspects of the guide and obtained suggestions. In particular:

- **we asked users if and how the virtual guide had been of help during the visit**. Interestingly, users that positively rated the guide declared either that the guide had been useful as a navigation aid (in particular, at the beginning of the visit), or that they had preferred to make a first exploration by themselves, and later used the guide for its ability to present the museum items in a logical order.

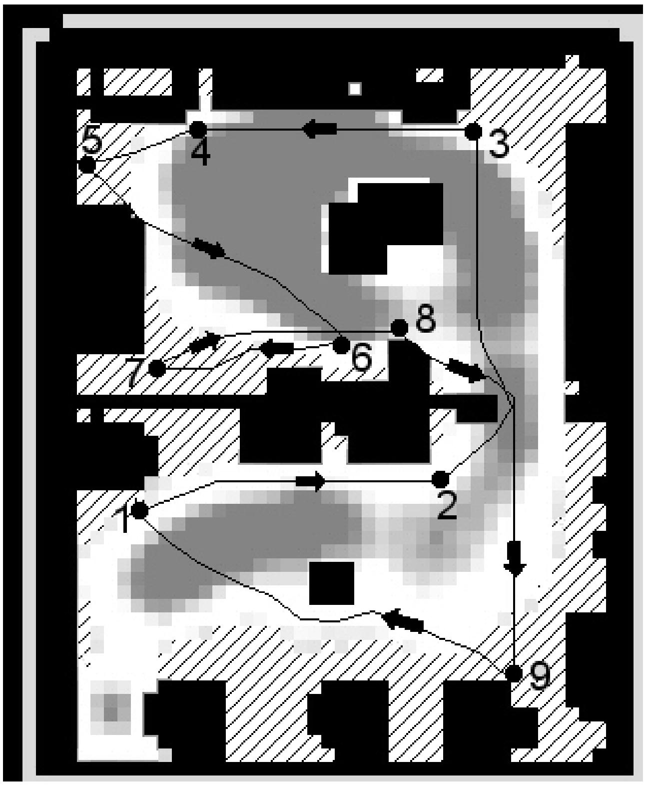


Fig. 8: Users' flows in the virtual museum during the evaluation; black areas represent museum building, items and furniture; white/grey represents areas visited by users (darker shades of grey correspond to areas of the environment where users spent more time), dots are locations of presentations of the virtual guide, lines connecting the dots are the paths followed by the virtual guide. The small dark grey area in the bottom left corresponds to the location where the visit started.

- **we asked users to rate the usefulness of the features of the virtual guide**. While many users simply said that the guide was useful, others complained about the walking speed of the guide (too slow or too fast), the talking speed of the guide (too slow or too fast), and some users were not satisfied with the appearance of the guide (some in terms of realism, e.g. lack of facial animation, and some in terms of preferred appearance, e.g. some would have preferred an aesthetically better agent).

we asked users to provide a wish list for the virtual guide. Some users pointed out that the guide was useful to first-time visitors, but they were not likely to use it in a second visit, unless it was able to provide a different tour. Other users pointed out that, in some cases, the presentations were boring, because they explained things they had previously learned by freely visiting the museum. Finally, a few users would have liked the ability for the guide to interact with museum items (e.g. open cabinets, push buttons, etc.).

User evaluation question	**1**	**2**	**3**	**4**	**5**
How would you define the interaction with the system? (1 – difficult, 5 – easy)	0%	0%	30%	40%	30%
How would you define the interaction with the system? (1 – boring, 5 – exciting)	0%	5%	45%	30%	20%
How would you rate the usefulness of the guide? (1 –useless, 5 – very useful)	0%	0%	15%	20%	55%
How would you rate the walking speed of the guide? (1 – slow, 5 – fast)	0%	20%	50%	20%	0%

Table 1: Evaluation results from the post-experiment questionnaire.

6. Conclusions

The results of the evaluation confirm other empirical studies (e.g. (Lester et al. 1997), (Towns, 1998)) on the positive effects that embodied agents have in the context of presentations, and in making a virtual place more lively and attractive for the user.

As a navigation aid, the embodied agent proved to be appreciated in the evaluation. It was mostly rated as simple to use, and it had the advantage of being unobtrusive for the expert user. Moreover, using the embodied agent demanded a very short learning time, probably because, from a human-computer interaction point of view, the guide metaphor has the advantage of being consistent with the real-world experience of users.

By analyzing the most frequent concerns expressed by subjects, there is a clear need for personalization capabilities. For example, walking and talking speed of the embodied agent were both rated too slow or too fast by some users: the ideal solution would be to adapt these features on the basis of each single user's preference. The visual appeal of the guide is also presumably dependent on each single user. Other concerns require more complex adaptation capabilities, e.g. the ability to adapt the explanations given in the tour on the basis of what the user previously did in the museum, or the ability to propose different tours in the case of multiple visits.

From this point of view, automating the process of deriving a guided tour for a 3D virtual environment opens up the possibility of building programs that dynamically generate different tours for different visitors, taking into account what is known about user's preferences, interests, and needs. In particular, the VRML Tour Creator can be integrated into any architecture that is able to dynamically generate VRML content, such as (Chittaro and Ranon, 2002; Walczak, 2002). The architecture we have recently proposed, called AWE3D (Chittaro and Ranon, 2002), explicitly supports the recording of usage data inside the VRML virtual environment and is thus well suited for personalization purposes.

With respect to future goals of this project, besides considering adaptivity, we plan to extend the possible functions and behaviors of the embodied agent. This ranges from the possibility of having users give high-level commands to the guide (e.g. "tell me more about this object", "I'm not interested in this object") to extending the guide abilities in interacting with the virtual environment, e.g. open doors and operate with interactive objects. Finally, we plan also to experiment the embodied agent approach in a wider virtual environment than the relatively small museum presented in this paper, thus exploring more deeply the relations between virtual environment size and use of the embodied agent.

7. Acknowledgements

Paolo Giangrandi played an important role in the design of the virtual museum application.

This work is partially supported by the MIUR COFIN 2003 program (project "User Interfaces for the Visualization of Geographical Data on Mobile Devices") and by the Friuli Venezia Giulia region (Regional Law 3/98, project "3D Web Sites for the Promotion of Tourism and Cultural Heritage").

8. References

Barbieri T., Paolini P. (2001). Reconstructing Leonardo's Ideal City From Handwritten Codexes To WebtalkII: A 3D Collaborative Virtual Environment System. *Proceedings of VAST 2001: Conference on Virtual reality, archeology, and cultural heritage*, Glyfada, Nr Athens, Greece. ACM Press, New York, 61-66.

Brusilovsky, P. (2001). *Adaptive hypermedia*. User Modeling and User Adapted Interaction, 11, (1-2), 87-110.

Chittaro, L., and Ranon, R. (2002). Dynamic Generation of Personalized VRML Content: a General Approach and its Application to 3D E-Commerce. *Proceedings of Web3D 2002: 7th International Conference on 3D Web Technology*, Tempe, Arizona, USA. ACM Press, New York, 145-154.

Chittaro L., Ranon R. and Ieronutti L. (2003). Guiding Visitors of Web3D Worlds through Automatically Generated Tours. *Proceedings of Web3D 2003: 8th International Conference on 3D Web Technology*, Saint Malo, France. ACM Press, New York, 27-38.

Chiu, M. L., Lin, Y. T., Tseng, K. W. and Chen, C. H. (2000). Museum of Interface. *Proceedings of The Fifth Conference on Computer-Aided Architectural Design Research in Asia* (CAADRIA 2000), Singapore. 471-480.

Cruz-Neira C., J. Sandin D., A. DeFanti T. (1993). Surround-screen projection-based virtual reality: the design and implementation of the CAVE. *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, Anaheim, CA, USA. ACM Press, New York, 135-142.

Darken, R.P., and Sibert, J.L. (1996). Wayfinding Strategies and Behaviors in Large Virtual Worlds. *Proceedings of CHI '96*, Vancouver, British Columbia, Canada. ACM Press, New York, 142-149.

Ennis, G., and Maver, T. (2001). Visit VR Glasgow – Welcoming multiple visitors to the Virtual City. *In Proceedings of eCAADe 2001: Education for Computer Aided Architectural Design in Europe*, Helsinki, Finland. 423-429.

Galyean T. (1995). Guided Navigation of Virtual Environments. *Proceedings of the 1995 symposium on Interactive 3D graphics*, Monterey, CA, USA. ACM Press, New York, 103-106.

Hendricks, Z., Tangkuampien, J. and Malan, K. (2003). Virtual Galleries : Is 3D Better?. *Proceedings of the 2nd international conference on Computer graphics, virtual Reality, visualisation and interaction in Africa*, Cape Town, South Africa. 17-24.

Ieronutti L., Ranon R. and Chittaro L. (2004). Automatic Derivation of Electronic Maps from X3D/VRML Worlds. *Proceedings of Web3D 2004: 9th International Conference on 3D Web Technology*, Monterey, California, USA. ACM Press, New York, 61-70.

International Standard ISO/IEC 14772-1:1997: The Virtual Reality Modeling Language. (1997) http://www.web3d.org/Specifications/VRML97/

Latombe, J.C. (1991). *Robot Motion Planning*, Boston, Kluwer Academic Publisher.

Lester, J., Converse, S. A., Stone, B. A., and Kahler, S. E. (1997). Animated Pedagogical Agents and Problem-Solving Effectiveness: A Large-Scale Empirical Evaluation. *Proceedings of the Eigth World Conference on Artificial Intelligence in Education*, Kobe, JAPAN. 23-30.

Rickel, J., and Lewis Johnson, W. (2000). Task-Oriented Collaboration with Embodied Agents in Virtual Worlds. In J. Cassell, J. Sullivan, and S. Prevost (Eds.), *Embodied Conversational Agents*. MIT Press, Boston.

Stoakley, R., Conway, M. J., Pausch, R. (1995). Virtual Reality on a WIM: Interactive Worlds in Miniature. *Proceedings of CHI 95*, Denver, Colorado, USA. ACM Press, New York, 265-272.

Thorndyke, P.W., and Hayes-Roth, B. (1982). Differences in Spatial Knowledge Acquired from Maps and Navigation. *Cognitive Psychology*, 14, 560-589.

Towns S. G., Callaway C. B., Voerman J. L., Lester J. C. (1998). Coherent Gestures, Locomotion, and Speech in Life-Like Pedagogical Agents. *Proceedings of IUI '98*, San Francisco, California, USA. 3-20.

van Mulken, S., André, E., and Muller, J. (1998). The Persona Effect: How Substantial is it? *Proceedings of HCI'98*, Sheffield, England. Springer Verlag, Berlin, 53-66.

Vinson, N.G. (1999). Design Guidelines for Landmarks to Support Navigation in Virtual Environments. *Proceedings of CHI '99*, Pittsburgh, Pennsylvania, USA. ACM Press, New York, 278-284.

Walczak, K. (2002). Building Database Applications of Virtual Reality with X-VRML. *Proceedings of Web3D 2002: 7th International Conference on 3D Web Technology*, Tempe, Arizona, USA. ACM Press, New York, 111-120.

Wernert, E. A. and Hanson, A. J. (1999). A framework for assisted exploration with collaboration. *Proceedings of IEEE Visualization '99*, San Francisco, California, USA. IEEE Computer Society Press, 241-248.

PsychNology Journal, 2004
Volume 2, Number 1, 43 - 60

SAMIR: A Smart 3D Assistant on the Web

F. Abbattista♠, G. Catucci, G. Semeraro and F. Zambetta

Dipartimento di Informatica – Università di Bari

ABSTRACT

A current trend in modern HCI is represented by Embodied Conversational Agents (ECAs), even designed to run on the Web. They are virtual 3D human-like front ends coupled with software agents that are able to engage in a conversation with a user and execute complex tasks, such as, for example, searching for some specific information or ordering some items from the catalogue of an online shop.
This paper presents SAMIR system, a framework to build intelligent agents for the Web.
SAMIR consists of a 3D face which is animated to exploit expressions which are perceived by the user; a custom version of the ALICE chatterbot to chat with the user; and finally an XCS classifier system to deal with the problem of keeping conversation and face expressions coherent with each other. Experimental results, taken from an online bookstore-based scenario, are presented at the end.

Keywords: *Conversational interfaces, intelligent agents, facial animation.*

Received 16 January 2004; received in revised form 7 April 2004; accepted 12 April 2004.

1. Introduction

People's life is rapidly shifting towards "new virtual dimensions", as opposed to the traditional physical one, where users can interact and communicate through the use of e-mail, Web, and chat technologies.

Different kinds of social relationships may be established in these virtual worlds and different interaction metaphors might be useful in order to enhance the sense of presence, i.e. the perception of "being there".

Immersive scenarios, such as the ones depicted in science fiction books like "Snow crash" (Stephenson, 2000) and "Neuromancer" (Gibson, 1995), introduced the idea that the World Wide Web becomes a totally interconnected 3D virtual environment, where people can retrieve information in a natural fashion, simply by navigating

♠ Corresponding Author:
Fabio Abbattista,
Dipartimento di Informatica, Università di Bari.
E-mail: fabio@di.uniba.it

through its “paths”, and people are embodied by 3D avatars, 3D animated representations of the user.

Based on these fascinating ideas, many research efforts are currently involved in developing Networked Virtual Environments (NVEs) (Singhal and Zyda, 1999), where people embodied by 3D avatars may interact with other people or autonomous computer-guided 3D characters.

This sort of autonomous characters, usually referred to as *Embodied Conversational Agents* (ECAs), are software components designed to act as virtual advisors within applications where a high level of human computer interaction is required.

Their aim is to overcome the classical limitations of the WYSIWYG interfaces, which cause inexpert users to suffer from an uncomfortable interaction experience.

Reactive and possibly pro-active embodied virtual personalities are able to understand users wishes, converse with them, find information and execute non-trivial tasks, replacing buttons pressing, menu choices and hyperlinks clicks, which often contribute to undesired information overload.

The metaphor of a face-to-face conversation greatly increases the feeling of presence during the interaction and eliminates the need of learning where to find specific widgets accomplishing a single task that is really needed.

Moreover, these systems are coupled with an animated 2D/3D look-and-feel, embodying their intelligence via a face or an entire body. This way it is possible to enhance user trust and satisfaction, giving users some sort of illusion of life, as cartoons, videogames or animated movies are able to do.

A very complete example of an embodied conversational agent is the REA system, developed in the Gesture and Narrative Language Group at the MIT Media Lab (Cassell, Sullivan, Prevost and Churchill, 2000). REA is an embodied conversational interface agent implementing conversational protocols in order to make interactions as natural as face-to-face conversations with another person.

REA acts as a real estate salesperson, answering user questions and showing users some virtual houses; it has an articulated graphical body and can sense the user passively through cameras and audio input; moreover, speech with intonation, facial display and gestures add a lot of depth to the ongoing interaction.

REA is rendered on a projection screen and the user stands in front of it: Two cameras mounted on top of the screen track user positions in space.

Users wear a microphone for capturing speech input, whilst an SGI Octane computer runs the graphics and conversation engine of REA, and several other computers manage the speech recognition and generation.

REA is able to conduct a conversation describing the features of the task domain while also responding to the user verbal and non-verbal input.

REA's responses are generated by an incremental natural language generation engine developed by Stone and Doran (1997), extended to synthesize gestures synchronized with speech output (Cassell, Sullivan, Prevost and Churchill, 2000).

Even though REA is a very complete ECA, it relies on a bunch of custom hardware devices for sensing user actions and requires a lot of computational resources (as many computers are used to run it).

Thus, important as it is as a proof of concept, REA cannot absolutely be defined as a portable system.

Egges, Kshirsagar and Thalmann (2003) describe a prototype system that uses the OCEAN model of personality (Costa and McCrae, 1992) and the OCC model of emotions (Ortony, Clore and Collins,1988) in combination with a dialogue system and a talking head with synchronised speech and facial expressions.

An ECA named Julie is described in a simple conversational context: Julie has to carry some boxes upstairs, so that the user might offer to help her in this tedious and tiring task.

Depending on the way Julie's personality is modelled (i.e., she is introvert or extrovert), her reactions vary from warm and kind thanks to shy and reluctant expressions and answers.

The centre of the application is a dialogue system, using Finite State Machines (FSM), which generates different responses based on the personality, mood and emotional state.

A visual front-end uses the dialogue output to generate speech and facial expressions, relying on MPEG-4 Facial Animation Parameters (MPEG-4 Committee, 2000).

The dialogue system annotates its outputs with emotional information, so that answers are coupled with expressions.

An example of such a response is (the tag |JO58| represents a joy emotion to a degree of 58%):

|JO58|Thanks! I like you too!

Even though the example application is very simple and it is based upon FSM, nevertheless it is remarkable for including both a personality model and a mood one.

The author of the Interface Project (Pandzic, 2001) implemented a full 3D MPEG-4 compatible facial animation system based on the Shout3D technology. An applet plays MPEG-4 FBA bitstreams, which are streamed from the server in real time. The audio track is streamed, as well.

Currently, fairly simple facial models are used, but the possibility of creating models and animation rules in a popular commercial 3D software package, like 3DS Max, gives the possibility of potentially using more complex models, due to the .S3D format exporter bundled with Shout3D.

The applet is controlled from Web pages by Javascript, making all interactions possible, and the virtual character can also bring up Web content in other frames of the Web page, which is synchronized with its talk.

These features enable a smooth integration in a Web site, whilst limitations include the size of the applet, which is something like 200K, as well as an unstable audio delivery.

Finally, no reasoning or learning schema is supplied with the system, so that it cannot be considered autonomous or adaptive under any point of view.

The MS Office assistants deserve to be mentioned in the agent panorama because of their widespread use, even though they are quite shallow in some respects. Indeed, they are too invasive and do not exhibit a very complex behavior. Nonetheless, they suffice to help the inexperienced user in many situations. They are based upon the Microsoft Agent technology (Microsoft Agent website http://www.microsoft.com/msagent/default.asp), a set of programmable software services that supports the presentation of interactive animated characters within the Microsoft Windows interface. Developers can use characters as interactive assistants to introduce, guide, entertain or enhance their Web pages or applications, in addition to the conventional use of WYSIWYG controls.

In addition to mouse and keyboard input, Microsoft Agent includes optional support for speech recognition, which allows applications to respond to voice commands using synthesized speech, recorded audio or text in a cartoon word balloon.

Another example of a Microsoft Agent is Instant Messaging Personalities (IMP), which gives a very good idea of how an agent can help users in handling all Microsoft Messenger features, such as mail checking, instant messaging and so on.

Agents have lips-synced faces pronouncing the messages the user receives and they tell the user whether any of his/her contacts has just gone online, offline, away, etc.

The system JackMOO (Cassel, Sullivan, Prevost and Churchill, 2000), developed at the Center for Human Modeling and Simulation, features *Smart Avatars* that understand actions they are asked to perform through the representation of actions, objects, and agents. This capability makes them a suitable solution for the execution of animations, as well as for a natural language expression.

A Parameterized Action Representation (PAR) is defined in order to drive a simulation in a given context of objects, agents and environment, and to support the considerable range of expression, nuance, purpose and manner offered by natural language.

JackMOO combines Transom Jack software with an underlying multi-user, network-accessible system that has been used for the construction of text-based conferencing, education, training and other collaborative software.

This system is a good model for virtual environment training applications, where several individuals can share a 3D space and learn team tasks and job coordination.

In this paper, we present the SAMIR (Scenographic Agents Mimic Intelligent Reasoning) system, a digital assistant where an artificial intelligence based Web agent is integrated with a purely 3D humanoid, robotic or cartoon-like layout. The outline of the paper is as follows: Section 2 presents the architecture of the SAMIR system, while sections 3, 4 and 5 detail the main modules of the system. An overview of the experimental results is given in section 6, whilst section 7 describes some conclusions and future work.

2. The Architecture of SAMIR

Figure 1 shows the client/server architecture of SAMIR. The overall behavior of the system is given by the interaction of three main subsystems disposed on one of the two sides of the application (Client Side and Server Side): The Behavior Manager (Abbattista, Paradiso, Semeraro, and Zambetta, 2004), the Dialogue Management System (DMS) and the Animation Module.

The communication between the client side and the server side is obtained through the common http protocol. The interaction with the system is delegated to the DMS client which captures user input and directs it to the DMS server and to the Behavior Manager. The DMS server selects the correct answer to each user request and forwards it to the DMS Client. The Behavior Manager covers another important aspect of this interaction: It creates the appropriate set of parameters encoding the emotional expressions of the system. The Animation Module uses these values to represent the corresponding facial expression, constructed by means of the various morph targets (Fleming and Dobbs, 1998) allowed by the system. In particular, some high-level morph targets are used in compliance with the six basic emotions as they were described by Ekman (Ekman, 1982), but even low-level ones are a feasible choice in order to preserve a full MPEG-4 compliance. The final step of this process performs a time varying key-frame interpolation, in order to animate the current facial expression.

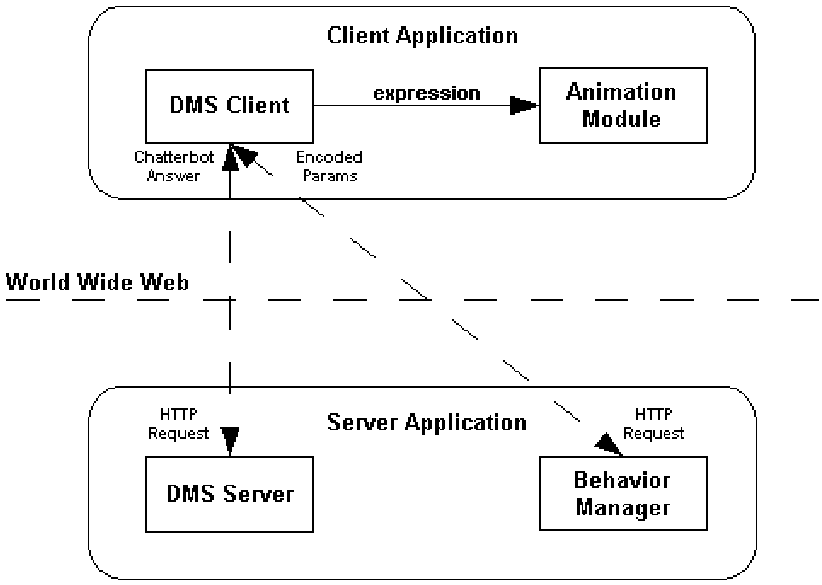


Fig. 1: The Architecture of SAMIR.

3. The Animation Module

The FACE (Facial Animation Compact Engine) Animation System is based on the work done to design and implement the Fanky Animation System (Paradiso, Zambetta and Abbattista, 2001). The Fanky system, in turn, solved some limitations of the Tinky system (Paradiso, Nack, Fries and Schuhmacher, 1999), such as the rendering limitations, due to the use of the VRML2.0 standard, and the need of installing a plugin to render 3D faces.

Our experience in Fanky led us to adopt the Shout3D API when implementing FACE: Good performances and a lot of new rendering capabilities were available since then, both in a hardware accelerated modality using the OpenGL API, and in a more limited fashion using a pure-Java software mode.

Even though Shout3D is almost compatible with VRML2.0, thanks to its extended capabilities we were able to create our 3D faces in the famous FaceGen software (http://www.facegen.com), to import them in 3D Studio Max and finally to export a single file in the Shout3D file format, containing both geometric and animation data.

However, FACE is more than a simple animation player because it relies on the idea of SACs (Standard Anatomic Components): Face regions in this model act as objects, in an object-oriented sense of the term, so that we can use different services for each of them.

Services range from reshaping, which exploit some Free Form Deformation algorithms (Sederburg, 1986), to low-level deformations such as FAPs (Facial Animation Parameters).

Moreover, SACs enable us to adopt different numerical methods to perform face region animations: Free Form Deformation, Waters muscle model and linear or non-linear key-frame interpolation schemes.

After having tried out all those methods, we decided to use by default a linear interpolation scheme based on morph targets, which is very well supported in the Shout3D API by means of the *ChannelDeformer* node because it is the best trade-off between rendering accuracy and speed.

FACE supports a variable number of morph targets in order to further improve performances: Either 12 high-level targets are used or the entire “low-level” FAP set, to achieve a full MPEG-4 compliance.

A small set of high-level parameters is very helpful when having to debug and improve the behavioral module because, of course, reasoning in terms of face expressions is better than having to deal with long streams of numerical low-level parameters, as for MPEG-4 animations.

However, many applications (e.g, teleconferencing, computer supported cooperative work, etc.) require MPEG-4 compliance and this led us to provide the possibility in case FACE, or the entire SAMIR system, has to be used by one of them.

FACE can use an unlimited number of timelines for rendering simultaneously both “synchronous” expressions, which result from conscious actions taken by the conversational agent, and “asynchronous” ones, due to the non-conscious reflexes commonly visible in every human face (i.e., eye-lid reflexes, small head movements, etc.). Figure 2 depicts some expressions taken from an animation stream.

This way an higher degree of realism is achieved, since the user feels to find the 3D agent more familiar and more similar to a human counterpart.

Finally, we are currently developing an in-house tool to perform not only the tasks accomplished by FaceGen, but to give the possibility of obtaining new 3D faces by applying some kind of free form deformation techniques.

The former modality is usable even by inexperienced users, whilst the latter is dedicated to users with a good experience in 3D modeling and 3D digital content creation.

Fig. 2: Some expressions assumed by a 3D face.

4. The Dialogue Management System

Two software modules, the “DMS Client” and the “DMS Server”, are the main components of the Dialogue Management System (DMS), a client/server application capable of managing user inputs and properly process them in order to obtain an adequate response to user requests and needs. The DMS Client adopts the Java applet technology to run in a WWW browser, and to capture user inputs to be sent to the DMS Server via the common http protocol. This module can also direct precise queries to the Web site of interest, relying on the information processed by the server counterpart. Such a goal is achieved through the use of Javascript technology embedded in the most popular WWW browsers available today.

The DMS Server analyzes the current input text typed by the user and forwarded by the DMS client, and looks for certain classes of words that are particularly relevant in the domain addressed by the system. The result of the task performed by the server side of the DMS consists of a textual response containing the result of the search issued by the user, or an invitation to be more precise in formulating requests. This task is accomplished through the use of the ALICE chatterbot, an open source software released by the ALICE AI Foundation, and the related Artificial Intelligent Markup

Language (AIML), an XML compliant language which provides a set of custom tags able to represent the vast set of possible user inputs, and to reduce them to a minimal collection of properly selected categories. All the knowledge of the system is entirely enclosed in the AIML dialogues, which are XML files storing the various categories cited above. An AIML category is an elementary unit structured in two main parts: A "pattern" section matching user input, and a corresponding "template" section containing an adequate response to user requests and/or actions (i.e., a JavaScript execution).

The communication with the client module has been obtained by means of the integration of the ALICE classical server engine in the SAMIR system as a Java servlet, which provides the necessary software libraries to handle the communication with the client.

Making a system able to assist users during navigation in a bookshop Web site required the definition of a set of AIML categories specifically tailored for book searching and shopping. The different categories implemented in the DMS system were chosen as close as possible to the way people request books in a real world bookstore. A set of seven fields was considered to let users specify the books he/she is interested in. They include the *book title, author, publisher, publication date, subject, ISBN code* and a more general field *keywords*.

Successful examples of book requests issued to the Amazon bookshop Web site follow:

- *I want a book written by Sepulveda*
- *I am looking for a book entitled Journey and whose author is Celine*
- *I am searching for all books written by Henry Miller and published after 1970*
- *I am interested in a book about horror*

or, in alternative forms, it is possible to send requests like these:

- *Could you find any book written by Fernando Pessoa?*
- *Search all books whose author is Charles Bukowski*
- *Give me the book whose ISBN code is 0-13-273350-1*
- *Look for some book whose subject is fantasy*

Not all user requests can be properly processed by the different AIML categories and, even more, it becomes very critical for the system to deal with those requests that exhibit a high level of ambiguity, due to the intrinsic fuzziness of the human language.

5. The Behavior Manager

The environment in which SAMIR acts is represented by the user dialogue (the higher the user satisfaction, the higher the reward received by SAMIR). Dialogues and facial expressions represent the communicative acts of our agent. The consistency between the facial expression of the character and the conversation tone is one of the main requirements of SAMIR.

At the very beginning of its *life*, the behavior of SAMIR is controlled by a set of random generated rules and, consequently, it is very stereotyped. By interacting with users, SAMIR is able to learn better behavioral rules in order to achieve even better performance.

To meet this goal, the learning module of SAMIR, namely the Behavior Manager, has been implemented through an XCS (Wilson, 1995), a machine learning system closely related to the Q-learning, but able to generate task representations which can be more compact than tabular Q-learning (Watkins, 1989). At discrete time intervals, the agent observes a state of the environment, takes an action, observes a new state and finally receives an immediate reward.

The learning process of an XCS includes three phases. In the acting phase, the performance component of the XCS selects the action to be performed. As a consequence, the system receives a feedback from the environment, which determines the effectiveness of the performed action. In this phase, the reinforcement component of the XCS evaluates the received reward and assigns a fitness value to the action chosen in the first phase. This process is iterated until the system is able to select a very fitted rule. In case of degrading performance (i.e., the reward from the environment is a negative value), the discovery component of the XCS starts an evolutionary process to generate new behavior rules. At the end of the evolution, the newly discovered rules substitute the less fitted rules of the XCS and the process is re-executed from phase 1. In SAMIR, behavioral rules are expressed in the classical format if <*condition*> then <*action*>. The <*condition*>, representing the state of the environment, is a combination of 4 possible events, sensed by 4 effectors, corresponding to different conversation tones such as: User salutation (user performs/does not perform salutation), user request formulation to the agent (no request, polite, impolite), user compliments/insults to the agent (no compliment, a compliment, an insult, a foul language), user permanence in the Web page (user changes/does not change the page). The <*action*> represents the expression that the Animation System displays during user interaction. Specifically, the expression is built as a linear combination of a set of basic expressions. We decided to extend the set of fundamental expressions proposed by Paul Ekman (anger, fear, disgust, sadness, joy, and surprise) (Ekman, 1982), by including some typical human expressions such as bother, disappointment and satisfaction. Thus the <*action*> part provides the Animation System with the percentage of each expression, to be used to compose the desired final expression of our character. For example, an expression composed by 40% of joy and 60% of surprise is coded into the following binary string:

0100	**0000**	**0110**	**0000**	**0000**	**0000**	**0000**	**0000**	**0000**
% of Surprise	% of Sadness	% of Joy	% of Fear	% of Disgust	% of Anger	% of Bother	% of Disappointment	% of Satisfaction

6. Experimental Results

In a preliminary phase, we defined a set of 30 interactional rules in which different situations have been considered (performed actions, user requests, etc.). This set of pre-defined rules represented the training set, which is the minimal know-how SAMIR should possess to start its *work* in the Web site.

To evaluate the performance of the system, during the training phase, the following criterion has been adopted for the credit assignment: A perfect match between the expected action and the action performed by the system is the best result, otherwise the credit assigned to the rule is inversely proportional to the distance between the expected action and the action performed by the system.

The experiment aims at verifying the effectiveness of the User Interaction module when learning the 30 predefined rules. We performed 5 runs of the system with the same parameters (see Table I) but with a different initial population. The values of the parameters have been empirically selected.

On average, the system was able to learn 26 out of 30 rules. The 4 unlearned rules were, in most cases, very close to the original predefined rules. The most interesting result has been the ability of the system to discover new rules, not included in the original set of 30 rules.

Parameter	Value
Maximum Population Size (N)	24000
Crossover Probability (Px)	1.0
Mutation Probability (Pm)	0.03
Learning Rate (β)	0.1
'Don't care' symbol Probability (P#)	0.05

Table 1: The parameters used in the first experiment.

For the sake of brevity, we omit to report the whole set of learned rules. Table II shows only some rules learnt by the system. The second and the third columns represent, respectively, the input to the system (the <*condition*> part of the rule) and the expected output (the <*action*> part of the rule). The fourth column shows the system output.

The first row in Table II represents a perfect match between the expected action and the action performed by the system. In the second row, the rule does not represent a perfect match but the two actions (the expected one and the action performed by the system) are quite *similar*. Last but not least, the third rule represents a new rule discovered by the system.

Due to the inherent features of the XCS, SAMIR was able to learn the pre-defined rules of behavior quite effectively and to generalize some new behavioral pattern that could update the initial set of rules. In such a way, SAMIR is comparable with a human assistant who, after a preliminary phase of training, will continue to learn new rules of behavior, based on personal experiences and interactions with human customers.

Rule	Input	Expected Output	System Output
1	0001001001100	0 0 0 0 60 40	0 0 0 0 60 40
2	0001111011000	0 0 60 0 0 0	0 0 40 0 0 0
3	0101100001001 (A novice user makes few errors)	-	70 0 30 0 0 0

Table 2: Results of the training phase

Having experimented the capability of the leaning module, we started a set of experiments aiming at verifying the effectiveness of SAMIR. We had to analyze two different characteristics of the system:

1. *User likeness*: Do users like/dislike to interact with a 3D animated face? Do users feel comfortable when interacting in natural language with SAMIR?

2. *SAMIR performance*: Is SAMIR able to interpret correctly books requests issued by a user? Do search results provided by SAMIR fit users requests?

Concerning the first issue, we have not yet performed a complete usability test. However, we asked some users, very acquainted with Web search engines, to record their first impressions on SAMIR. In the experiment, users were asked to interact with SAMIR for a week. In the first part of the week they were free to converse with the agent on general topics like sports, politics, etc. In the last three days, users were forced to ask SAMIR for some books on the Amazon.com website. At the end of the week, users were asked to fill a questionnaire and to report their comments and suggestions about their experience.

The results of this preliminary test show that, on average, users like to converse with an animated 3D agent (most of our users thought of SAMIR as funny, interesting, etc.). On the other hand, they do not think SAMIR is useful in searching books and they still prefer to use the classical Amazon.com search engine. This result is not surprising since our users access Web search engines (such as Google) on a daily basis and are quite expert in composing effective queries. In the next future we will perform a complete test involving more users with different skill levels, to collect a more significant sample of users reactions to SAMIR behavior.

In the rest of this section we present some examples obtained from the interaction between SAMIR and some typical users, searching for books about topics like literature, fantasy and horror or looking for books that match specific constraints concerning title, author and publisher are given. As can be seen in the following figures, SAMIR has been able to find books satisfying user requests.

First of all, in Figure 3, we take a look at the initial presentation of the SAMIR Web Agent as it starts up; SAMIR presents itself and asks the user his/her name for user authentication and recognition.

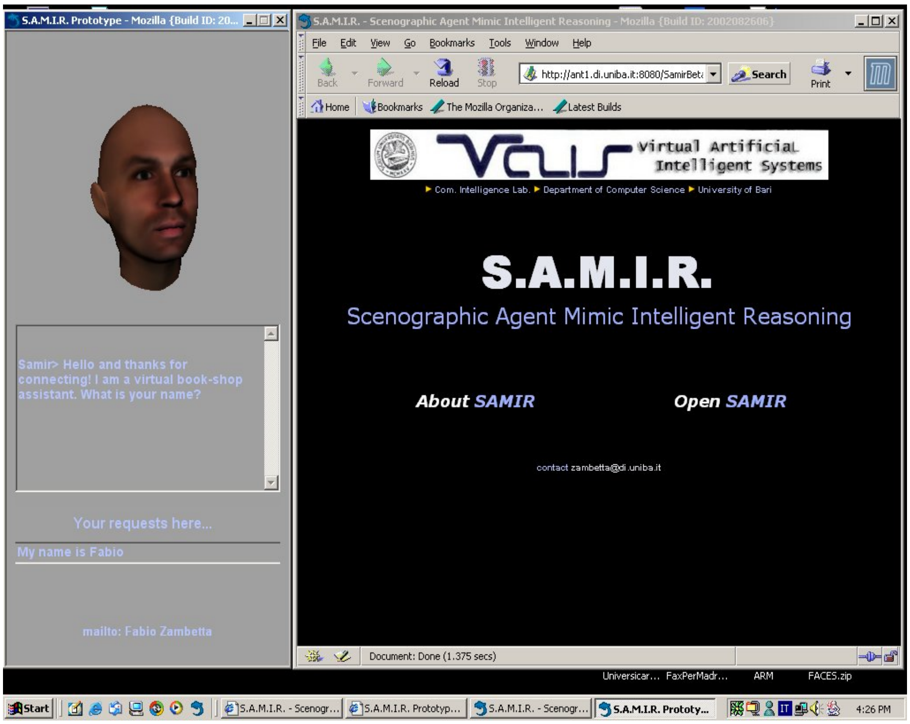


Fig. 3: **SAMIR log.**

Figure 4 shows the results of a user request about a horror book. The result of the query is a set of books in this genre, available at the Amazon book site.

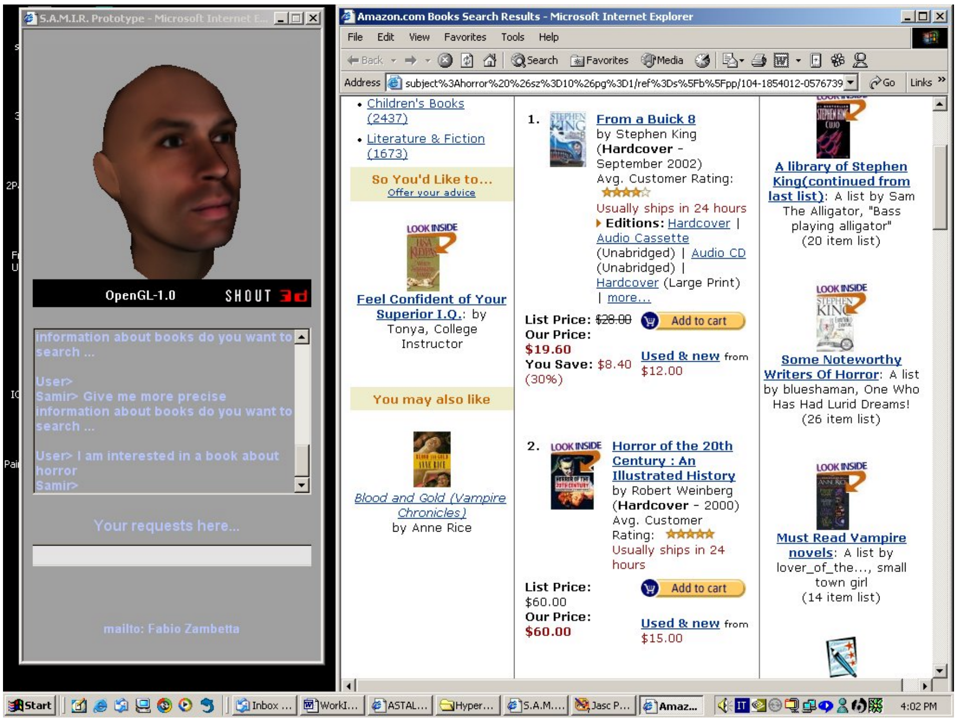


Fig. 4: Requesting an horror book.

Figure 5 shows a more specific query of a user interested in a list of books by the author Sepulveda.

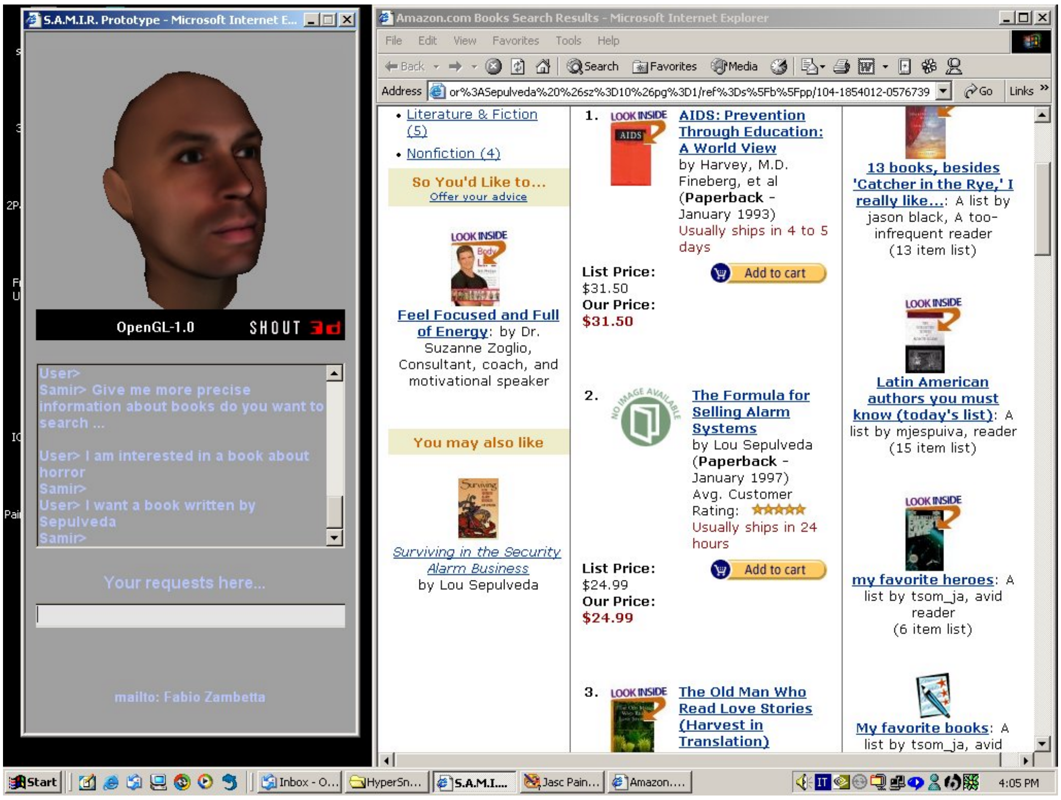


Fig. 5: Results about Sepulveda author.

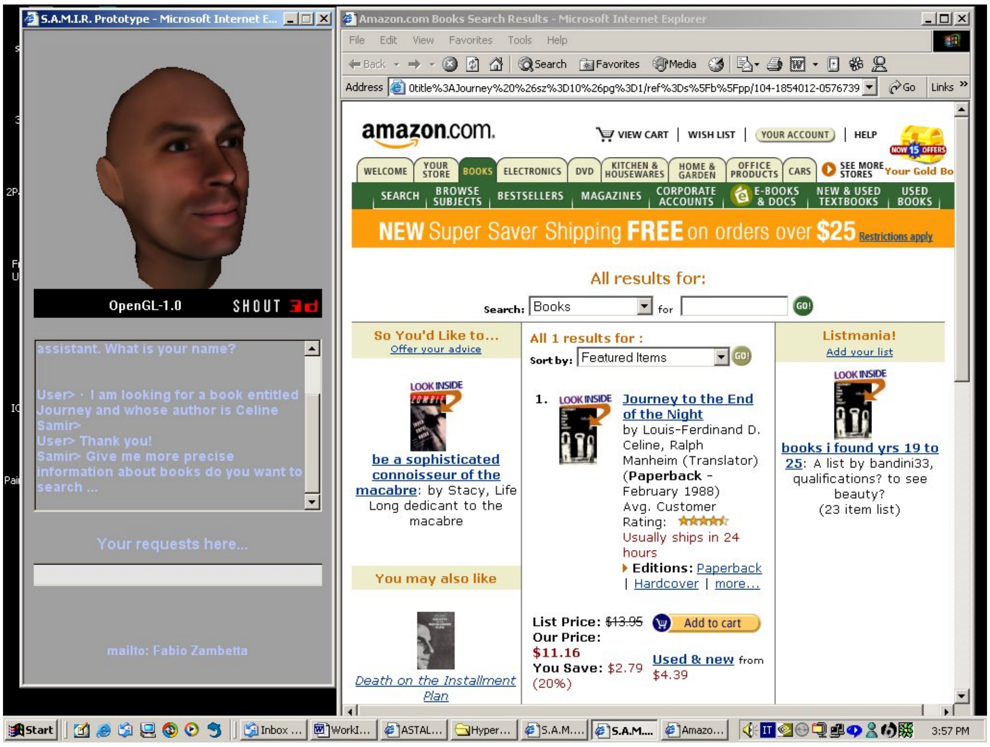


Fig. 6: Journey to the End of the Night by Celine

Figure 7 is an example of a more sophisticated query in which there is a request for Henry Miller books published after 1970. In this case the user gives a heavy insult to SAMIR and, consequently, its expression is angry.

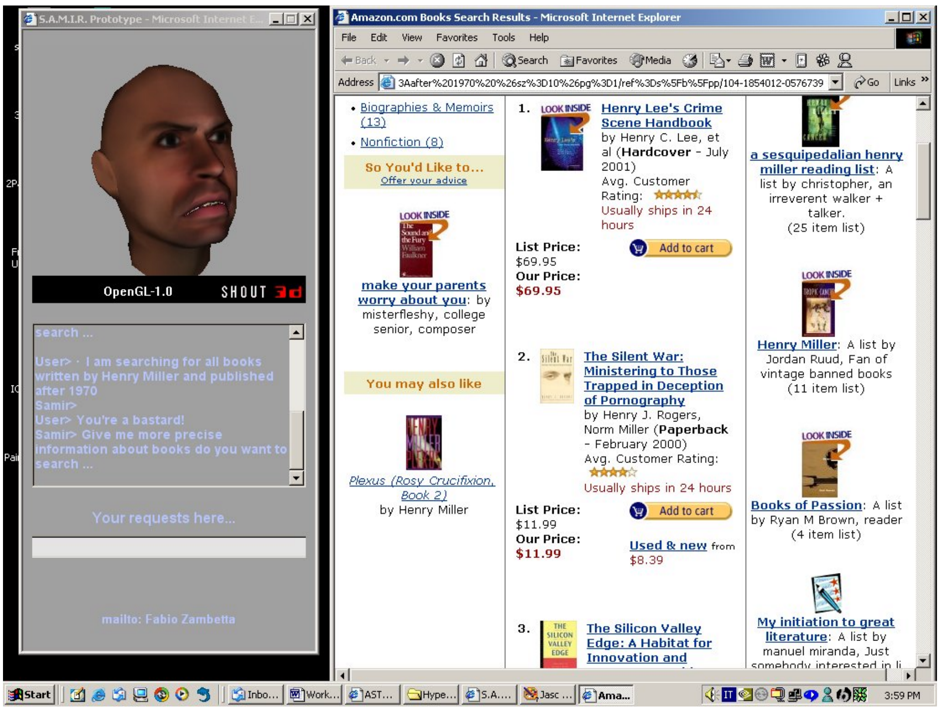


Fig. 7: All books by Henry Miller published after 1970.

7. Conclusions

In this paper we presented a first prototype of a 3D agent able to support users in searching for books in a Web site. The prototype has been linked to a specific site, but we are implementing an improved version that will be able to query several Web bookstores simultaneously and to provide users with a report that summarizes a comparison based on different criteria, such as price and delivery times.

The SAMIR system offers a wide covering of heterogeneous aspects (e.g., a 3D multi-modal interface, a learning module and a dialogue subsystem) in a very light system. SAMIR offers the possibility of being used via a common Java applet embedded in a Web browser and it will also be available as a stand-alone application which can use more rendering features. Moreover, the focus of the system is centered on creating a realistic type of interaction, by keeping the expressions of the 3D face emotionally coherent with the dialogue flow.

Our work will be further aimed at giving a more natural behavior to our agent. This can be achieved by improving dialogues, and possibly, the text processing capabilities of the ALICE chatterbot, and giving the agent a full proactive behavior: The XCS should be able not only to learn new rules to generate facial expressions but also to modify dialogue rules, to suggest interesting links and to supply an effective help during the site navigation.

8. References

Abbattista, F., Paradiso, A., Semeraro, G., and Zambetta, F. (2004). *An agent that learns to support users of a Web site*. Applied Soft Computing, 4(2004), 1-12.

Cassell, J., Sullivan, J., Prevost, S., and Churchill, E. (Eds.) (2000). *Embodied Conversational Agents*. MIT Press, Cambridge.

Costa, P.T., and McCrae, R.R. (1992) *Normal personality assessment in clinical practice: The NEO personality inventory*. Psychological Assessment, 4(1), 5–13.

Egges, A., Kshirsagar, and S., Thalmann, N. M. (2003). *A Model for Personality and Emotion Simulation*. In Palade, V., Howlett, R.J., Jain, L.C. (Eds.). Knowledge-Based Intelligent Information and Engineering Systems. Springer, Berlin, 453-461.

Ekman, P. (1982). *Emotion in the human face*. Cambridge University Press, Cambridge.

Fleming, B., and Dobbs, D. (1998). *Animating Facial Features and Expressions*. Charles River Media, Hingham.

Gibson, W. (1995). *Neuromancer*. HarperCollins, New York.

MPEG-4 Committee, (2000). *The MPEG-4 standard specification*, http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm.

Ortony, A., Clore, and G.L., Collins, A. (1988). *The Cognitive Structure of Emotions*. Cambridge University Press, Cambridge.

Pandzic, I.S. (2001). *Life on the web*. Software Focus Journal, 2(2), 52-59. John Wiley & Sons.

Paradiso, A., Nack, F., Fries G., and Schuhmacher, K. (1999). *The Design of Expressive Cartoons for the web – Tinky*. In Proc. of *ICMCS Conference*, IEEE Press, 276-281.

Paradiso, A., Zambetta, F., and Abbattista, F. (2001). *Fanky: A tool for animating 3D intelligent agents*. In de Antonio, A., Aylett, R., and Ballin, D., (Eds.). Intelligent Virtual Agents. Springer, Berlin, 242-243.

Sederburg, T.W. (1986). *Free-Form Deformation of Solid Geometric Models*. Computer Graphics, 20(4), 151-160.

Singhal, S., and Zyda, M. (1999). *Networked Virtual Environments: Design and Implementation*. Addison Wesley, Boston.

Stephenson, N. (2000). *Snow crash*. Bantam Books, New York.

Stone, M., and Doran, C. (1997). *Sentence Planning as Description Using Tree-Adjoining Grammar*, *ACL.* MIT Press, Madrid, 198—205.

Watkins, C.J.C.H. (1989). *Learning from delayed rewards*. *PhD thesis*, University of Cambridge, Psychology Department.

Wilson, S.W. (1995). *Classifier Fitness based on Accuracy*. Evolutionary Computation, 3(2), 149-175.

PsychNology Journal, 2004
Volume 2, Number 1, 61 - 83

Developing Affective Lexical Resources

Alessandro Valitutti[♦], Carlo Strapparava, Oliviero Stock

ITC-irst, Trento, Italy

ABSTRACT

Affective computing is advancing as a field that allows a new form of human computer interaction, in addition to the use of natural language. There is a wide perception that the future of human-computer interaction is in themes such as entertainment, emotions, aesthetic pleasure, motivation, attention, engagement, etc. Studying the relation between natural language and affective information and dealing with its computational treatment is becoming crucial. In this paper we present a linguistic resource for a lexical representation of affective knowledge. This resource (named WORDNET-AFFECT) was developed starting from WORDNET, through the selection and labeling of the synsets representing affective concepts.

Keywords: *Affective Computing, NLP, Lexical Resources, WORDNET.*

Received 20 January 2004; received in revised form 31 March 2004; accepted 1 April 2004.

1. Introduction

Affective computing is advancing as a field that allows for a new way of human computer interaction. In the most sophisticated cases, the interaction modality is meant to be based on language. There is a wide perception that the future of HCI is in themes such as entertainment, emotions, aesthetic pleasure, motivation, attention, engagement, etc. Studying the relationship between natural language and affective information and dealing with their computational treatment is becoming crucial. Initial user studies show that computers' affective ability plays a vital role in improving interaction with users. This ability depends not only on the affective expressiveness, but also on the capacity to detect the affective state of the user. Researchers have tried detecting the user's affective state in many ways, such as through facial expressions, speech, physiology, and text. In particular, text is an important modality for sensing affect because the bulk of computer user interfaces today are textually based. Examples of such applications are synthetic agents for giving affective responses to the user input at the sentence level (e.g. an affective text analyzer architecture [Liu et

♦ Corresponding Author:
Alessandro Valitutti,
ITC-irst, Via Sommarive 18,
38050, Povo, Trento, Italy.
Phone: +39 0461 314-568
Fax: +39 0461 302040
Email: alvalitu@itc.it

al. 2003], and affective text-to-speech systems). In other applications, the affective interaction is finalized to influence the emotional state of the user. We have worked in this direction, in particular addressing some aspects of computational humour. For instance, we have exploited NLP to build systems that were capable of inducing amusement and affecting the emotional state of users (e.g. HAHAcronym [Stock and Strapparava 2003]). For these applications it is necessary to have linguistic resources containing affective knowledge, which, unfortunately, cannot be found off-the-shelf. In order to develop a linguistic resource of this type, it is necessary first to consider the lexical components of the affective language.

In this paper we describe our efforts in this direction. We developed a preliminary version of a lexical knowledge base containing words in an affective lexicon connected with a set of affective concepts. This resource (named WORDNET-AFFECT) was developed starting from the lexical knowledge base WORDNET, through a selection and labeling of the affective concepts (represented by sets of synonyms). WORDNET-AFFECT was then extended taking into account OpenMind, a database of common sense sentences, in which there is a considerable amount of common sense knowledge [Singh et al. 2002]. Exploiting WORDNET-AFFECT and a word sense disambiguation algorithm [Magnini et al. 2002], we automatically chose an "affective-oriented" subset of OpenMind, named OpenMind-Affect. OpenMind-Affect is composed of sentences, patterns, and parsing trees containing affective concepts. In this way the affective lexicon is enriched by "contextual words", which do not directly refer to affective state (emotions and mood), but that are meaningful from the affect point of view.

2. Applicative relevance of affective lexicon

2.1 Affective verbal language and HCI

As claimed by Picard [1997], there are contexts in which human-computer interaction can be improved with communication that involves the use of emotional information. Affective communication can be performed with facial expressions, physiological and behavioral responses to the affective states, and natural language. We are particularly interested in the verbal component of the affective communication, for different reasons.

Regarding the applications in affect recognition, the nonverbal channel is insufficient for expressing the full range of human emotional experiences. According to Fussell, nonverbal cues can indicate what general class of emotions a person is feeling, but they typically do not provide detailed information about that person's emotional state. "By seeing that someone is crying, for instance, we might assume that they are sad; by the extent of sobbing we might even be able to infer the intensity of the sadness. But the tears in and of themselves provide no information about the particular experience of sadness. Instead, verbal descriptions of emotional states can provide quite precise information about the specific form of an emotion, such as anger, depression, or happiness, that a person is experiencing" [Fussell 2003]. The richness of natural language makes it possible to express emotions in different modalities, and allows us to distinguish explicit communication (as in *introspective reports*) from unintentional communication (e.g. when the form or the lexical content of the expression reveals an appreciative or depreciative disposition of the speaker).

These characteristics of affective verbal language allow us to develop systems for the detection and modelling of user affective states. The affective user modelling can be used to realize an "emotional intelligence" consisting of the ability to recognize and appropriately respond to human emotions [Picard 2000]. For these applications, the behaviour of the system is not describable only in terms of affect recognition or expression, but consists of a complex adaptive interaction with the user.

In several applicative contexts the verbal output of the system depends not only on the actual affective state of the user, but especially on the desired one. For example, in computational humour and computational persuasion [Guerini, Stock and Zancanaro 2003], the system tries to put the user in the desired mental state (which, in the case of computational humour, is *amusement*). In this case, the affective verbal language is actively used by the program, instead of by the user.

In all HCI applications in which it is necessary to have an affective interaction, it is very useful to have an affective lexical resource. On this subject, let us consider some existing applicative research and systems.

2.2 Example applications

Elliot's *affective reasoner*. This is a collection of Artificial Intelligence programs that reason about human emotion, and are embodied in multimedia computer agents. It was conceived and developed by Clark Elliot [1992], but it is originally based on the theoretical work of Andrew Ortony et al. [1988]. The model on which the system was developed consists of a collection of 26 emotion categories related to *eliciting conditions* (*events*, *objects and persons*, *actions*) through a set of rules. The conditions determine the choice of the emotion and a corresponding emotional response, e.g. a convenient facial expression, for an embodied agent, or a verbal utterance, for a conversational agent. For the latter, availability of organized lexical resource allows us to enhance the verbal expressivity.

Information and tutoring tools. These systems use natural language generation to provide information on a particular subject, or to instruct how to perform some complex action. There are domains in which it is useful to produce messages that are empathetic to the hearer. In this case, the form of the messages is as important as the content. For example, when the message content produces an emotional effect on the subject, the form may offset the “unpleasant” information and stressing the “favourable” one, through mitigating or enhancing terms (such as detensifier and intensifier adverbs) [De Rosis and Grasso 1999]. For this purpose, an affective lexical resource can provide a wide spectrum of lexical variants of the same concept, with different affective weights.

Affective text sensing systems. These are programs for assessing the affective qualities of natural language. They can be very useful for HCI systems performing text based affective user modeling. A new interesting approach, corpus-based, is that of Liu et al. [2003]. The affect of the text, at the sentence level, is classified into one of six basic categories of emotion. The analysis is performed through a model built starting from *OpenMind Commonsense*, a large-scale collection of common sense knowledge. Liu et al. chose a list of emotion words (named *ground words*) by which to bound a first set of affective sentences in OpenMind. These sentences contain other words on which the affective information of the ground words is propagated, with an *attenuation factor*. By these new words, a new set of affective sentences in OpenMind is individuated, and so on. This approach can be improved by increasing the number of ground words and by considering the senses of the words. Then, a lexical resource including the relation between affective words and concepts is required.

Computational humour. There are some situations where humour can play an important role in improving human-computer interaction (e.g. edutainment or frustration reduction). These are very difficult tasks, but there are some recent positive results in this direction. Stock and Strapparava [2003] have worked at a concrete limited problem for the core of the European Project HAHAcronym. The main goal of HAHAcronym has been the realization of an acronym ironic re-analyzer and generator. The re-analyzer takes as input an acronym with its expansion, and gives as output the same acronym with a humorous expansion. Making fun of existing acronyms amounts to basically using irony on them, desecrating them with some unexpected contrasting but otherwise consistent sounding expansion. In this system, ironic reasoning is developed mainly at the level of acronym choice and in the incongruity resulting in the relation to the coherently combined words of the acronym expansion. The acronym generator is more complex than the re-analyzer. In this case, the input is a concept from which the system generates both the acronym and the expansion.

The availability of an affective lexical resource can improve this strategy by allowing the system to focalize the incongruity at the affective level. For re-analyzing, a positive or a negative valence value is attributed to the acronym, and then the expansion generation must include affective words (e.g. appreciative and depreciative words) with opposite valence. For acronym generation, the valence opposition should be applied to both the input concept and the acronym.

3. The model for the organization of the affective lexicon

In order to organize the affective lexicon, it is necessary to have a model for the affective knowledge representing emotions, moods, attitudes, and traits. Using this model we should be able to identify a large number of affective concepts, organize them in a hierarchical structure and connect them via the lexicon. The past literature on affective lexicons guided us in the search of the more suitable model.

3.1 Limitations of the lexical semantics approach

First attempts to build a lexical structure for affective terms concerned studying which terms are really representing emotions, and what classification criteria to consider. In particular, lexical semantics approach was founded on the belief that “it is possible to infer emotion properties from the emotion words” [D’Urso and Trentin 1998]. This approach consists of three main steps. First, emotion words are collected from dictionaries [Weigand 1998] or from literary and newspaper texts. Then, a fixed number of semantic contexts are fixed: e.g. pure emotion terms, personality trait terms, physical and cognitive state terms, etc. [Ortony et al. 1987]. Finally, from each term a set of affective dimensions is extracted, using techniques such as *factorial analysis* [Nowlis and Nowlis 1956] or *multidimensional scaling* [Young et Hamer. 1987].

The lexical semantics approach showed a number of important issues. Ortony and Clore [1981] reviewed the literature on emotion labels, and they suggested that the process used to select emotion words has not led to a domain of emotion words exclusively (e.g. word "anger" refers to an emotion, "animosity" to a mood, and "confusion" to a cognitive state). Another problem was outlined by Watson and Tellegen [1985]: in the literature there is agreement only on some features such as "arousal" (*excited*, *tense* versus *relaxed*, *sleepy*) and "valence" (*happy*, *glad* versus *sad*, *upset*). However, these two dimensions are not sufficient to individuate the whole spectrum of emotional concepts. Moreover, the techniques of the lexical semantics approach (e.g. *factorial analysis* and *multidimensional scaling*) don't allow us to distinguish different senses of the same word. For example, the word *surprise* may refer to a *feeling* ("the astonishment you **feel** when something totally unexpected happens to you"), to an *event* ("a sudden unexpected **event**"), or to an *action* ("the **act** of surprising someone").

In our approach to the affective lexicon, the center of interest is not to study the nature of emotions, but how the affective meanings are expressed in natural language. Therefore, in order to build a structure for organizing the affective lexicon, we cannot use only information coming from the lexicon itself, but we need to get affective information from "extra-linguistic" sources, provided by recent scientific research on emotion. We preferred to look for an existing lexical resource and to enrich it with addictional affective information by a semiautomatic tagging procedure.

3.2 The adoption of WORDNET as a model for the affective concepts

The sought-after model must exhibit an explicit representation of the concepts, wide lexical coverage and a simple correlation between concepts and words. We think that the lexical knowledge base WORDNET would be the best candidate for satisfying these requisites. In the following section we provide an overview on WORDNET and on its structural organization.

4. WORDNET and WORDNET DOMAINS

As far as the representation and organization of lexical information is concerned, a key concept is the idea of a Lexical Knowledge Base (LKB), proposed, among others, by [Briscoe 1991] and [Calzolari 1992] to provide information, mostly of a semantic nature, consistently structured and available electronically. A Lexical Knowledge Base is an evolution both from a Machine Readable Dictionary, in which one finds an electronic version of the paper dictionary, and from a Lexical Data Base, in which part of the information available in the text format dictionary has been extracted.

One of the most significant attempts to realize a large scale LKB is WORDNET[1], a thesaurus for the English language based on psycholinguistics principles and developed at the Princeton University by George Miller [Miller 1990; Fellbaum 1998]. WORDNET organizes lexical information in terms of word meanings, rather than word forms. It has been conceived as a computational resource, improving some of the drawbacks of traditional dictionaries, such as the circularity of the definitions and the ambiguity of sense references. English nouns, verbs, adjectives and adverbs (about 130,000 lemmas for all the parts of speech in version 1.6) are organized into synonym classes (*synsets*), each representing one underlying *lexical concept* (about 100,000 synsets). Lemmas (about 130,000 for version 1.6) are organized in synonyms classes (about 100,000 synsets). WORDNET can be described as a "lexical matrix" with two dimensions: a dimension for *lexical relations*, that is relations holding among words and thus language-specific, and a dimension for *conceptual relations*, which hold among senses (in WORDNET they are called synsets) and that, at least in part, we consider independent from a particular language. In Table 1 an example of a lexical matrix is reported. *Word form* refers to the physical utterance or inscription; *word meaning* refers to a lexicalized concept. F_1 and F_2 are synonymous, while F_2 is polysemous. Polysemy and synonymy are problems gaining access to information in the mental lexicon.

	Word forms				
Word meaning	F_1	F_2	F_3	…	F_n
M_1	E_{11}	E_{12}			
M_2		E_{22}			
M_3			E_{33}		
…				…	
M_m					E_{mn}

Table 1: WORDNET Lexical Matrix.

A synset contains all the words by means of which it is possible to express the synset meaning: for example the Italian synset {`calcium, calcio, Ca`} describes the sense of "calcio" as a chemical substance, while the synset {`calcio, pedata`} describes the sense of "calcio" as a leg movement. Here a list of some relations present in WORDNET follows.

[1] WORDNET is freely available, for research purposes, at `http://www.cogsci.princeton.edu/~wn/`.

4.1 WordNet Relations

Synonymy. The most important lexical relation for WORDNET is the similarity of meaning, since the ability to recognize synonymy among words is a prerequisite to build synsets and therefore meaning representation in the lexical matrix. Two expressions are synonymous if substitutivity is valid (in other words if the substitution of one with the other does not change the truth value of a phrase). It is important to note that defining synonyms in terms of substitutivity requires partitioning WORDNET into nouns, verbs, adjectives and adverbs. This is consistent with the psycholinguistic evidence that nouns, verbs, adjectives and adverbs are independently organized in the human semantic memory. Obviously if a word pertains to more than one synset, this gives an indication of its polysemy.

Antonymy. This is another familiar relation among words. It provides the organizing principle for adjectives. The antonym of a word *w* in general is not-*w*. However there can be exceptions to this interpretation: for instance, while “rich” and “poor” are antonyms, the statement that someone is not rich does not implies that he is poor.

Hyperonymy / Hyponymy. This corresponds to the well known ISA relation. In a different way from synonymy and antonymy, hyperonymy (and its inverse hyponymy) is a relation between meanings, so it holds among synsets. As an example the synset {`apple tree`} is a hyponymy of the synset {`tree`}, which in turn is an hyponymy of {`plant`}. This relation provides the organizing principle for the noun hierarchy. Given a Hyperonymy/Hyponymy hierarchy it is possible to calculate the “coordinate-terms” for a given synset. For example, among the “coordinate-terms” for {`horse`} there are the synsets {`mule`} and {`zebra`}, which are common hyponyms of the synset {`equine, equid`}.

Meronymy / Holonymy. This represents the relation between a whole and its parts. It is a relation among synsets. Three types of homonymyc relations, along with their meronymyc inverse, are used in WORDNET: member-of (e.g. {`tree`} is member-of {`forest`}); part-of of (e.g. {`kitchen`} is part-of {`apartment`}); substance-of of (e.g. {`hydrogen`} is substance-of {`water` H_2O}).

Entailment. This is a semantic relation used for defining the verb hierarchy. From a logic point of view a proposition P “entails” a proposition Q if there is no state of the world in which P is true and Q is false. As an example the synset {`snore`} implies the synset {`sleep`}.

Troponymy. The entailment relation is at the base of the definition of the "troponymy" relation, which holds among verbs: in fact synset S1 is troponym of synset S2 if S1 implies S2 and if S1 is temporally co-extended with S2 (e.g. the synset {`walk`} is a troponym of the synset {`move`}).

ITC-irst has developed WORDNET DOMAINS, a multilingual extension of the well-known English WORDNET. This is a general tool: it is a multilingual lexical database where English and Italian senses are aligned. This multilingual extension [Artale et. al. 1997] [Magnini and Strapparava 1997] is based on the assumption that a large part of the conceptual relations defined for the English (about 72,000 ISA relations and 5,600 part-of relations) can be shared with Italian. From an architectural point of view, the Italian part of WORDNET DOMAINS implements an extension of the WORDNET lexical matrix to a "multilingual lexical matrix" through the addition of a third dimension relative to the language.

Some specific algorithms for use in computational humour have to be developed on top of WORDNET DOMAINS. A fundamental tool is an incongruity detector/generator: concretely we need to be able to detect or propose semantic mismatches between word meanings in the acronym context. This incongruity detector/generator exploits the domain augmentation described in the following section.

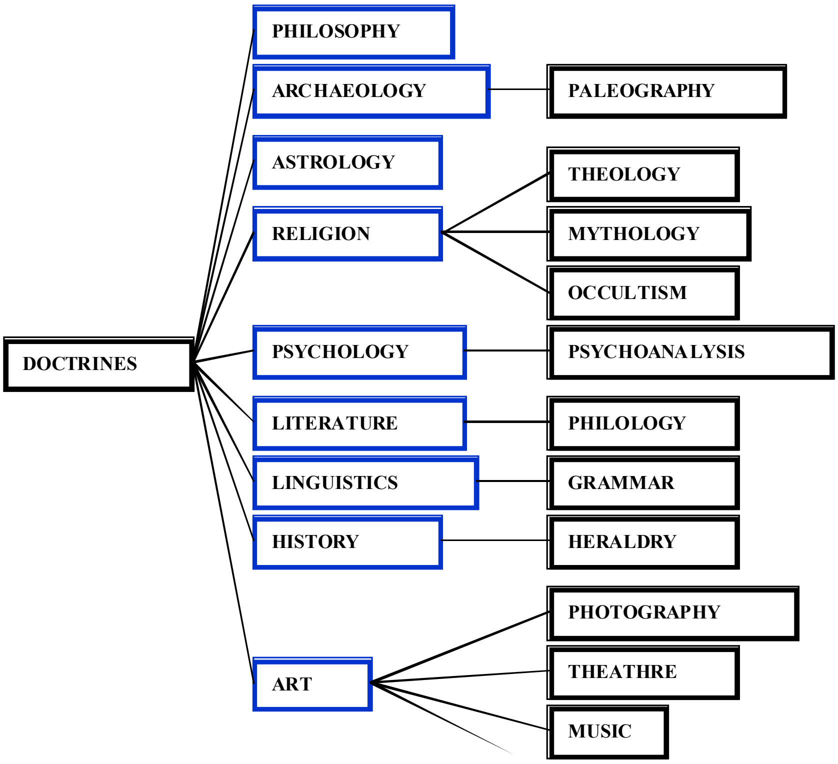


Fig.1: A portion of the domain hierarchy.

4.2 Augmenting WORDNET with Domain information

Domains have been used both in Linguistics (i.e. Semantic Fields) and in Lexicography (i.e. Subject Field Codes) to mark technical usages of words. Although this is useful information for sense discrimination, in dictionaries it is typically used for a small portion of the lexicon. WORDNET DOMAINS[2] is an attempt to extend the coverage of domain labels within an already existing lexical database, WORDNET (version 1.6). Synsets have been annotated with at least one domain label, selected from a set of about two hundred labels hierarchically organized. (Figure 1 shows a portion of the domain hierarchy).

[2] WORDNET DOMAINS and WORDNET-AFFECT are both freely available, for research purposes, at `http://wndomains.itc.it.`

Sense	Synset and Gloss	Domains
#1	depository financial institution, bank, banking concern, banking company (a financial institution…)	ECONOMY
#2	bank (sloping land…)	GEOGRAPHY, GEOLOGY
#3	bank (a supply or stock held in reserve…)	ECONOMY
#4	bank, bank building (a building. . .)	ARCHITECTURE, ECONOMY
#5	bank (an arrangement of similar objects…)	FACTOTUM
#6	savings bank, coin bank, money box, bank (a container…)	ECONOMY
#7	bank (a long ridge or pile…)	GEOGRAPHY, GEOLOGY
#8	bank (the funds held by a gambling house…)	ECONOMY, PLAY
#9	bank, cant, camber (a slope in the turn of a road…)	ARCHITECTURE
#10	bank (a flight maneuver. . .)	TRANSPORT

Table 2: WORDNET senses and domains for the word "bank".

We have organized about 250 domain labels in a hierarchy (exploiting Dewey Decimal Classification), where each level is made up of codes of the same degree of specificity: for example, the second level includes domain labels such as BOTANY, LINGUISTICS, HISTORY, SPORT and RELIGION, while at the third level we can find specialisation such as AMERICAN_HISTORY, GRAMMAR, PHONETICS and TENNIS.

Information brought by domains is complementary to what is already in WORDNET. First of all a domain may include synsets of different syntactic categories: for instance MEDICINE groups together senses from Nouns, such as `doctor#1` and `hospital#1`, and from Verbs such as `operate#7`. Second, a domain may include senses from different WORDNET sub-hierarchies (i.e. deriving from different "unique beginners" or from different "lexicographer files"). For example, SPORT contains senses such as `athlete#1`, deriving from `life_form#1`, `game_equipment#1`, from `physical_object#1`, `sport#1` from `act#2`, and `playing_field#1`, from `location#1`.

Finally, domains can have an important role in the design of Word Sense Disambiguation algorithms. In fact they may group senses of the same word into homogeneous clusters, with the side effect of reducing word polysemy in WORDNET.

Table 1 shows an example. The word "bank" has ten different senses in WORDNET 1.6: three of them (i.e. `bank#1`, `bank#3` and `bank#6`) can be grouped under the ECONOMY domain, while `bank#2` and `bank#7` both belong to GEOGRAPHY and GEOLOGY, causing the reduction of the polysemy from 10 to 7 senses. The ITC-irst particular approach to Word Sense Disambiguation (called Word Domain Disambiguation) has shown good results at the Senseval-2 competition [Magnini *et al.*2001].

5. WORDNET-AFFECT

Our work on affective lexicon was focused on the realization of a resource that contains the set of affective concepts correlated to the affective words. The availability of the WORDNET database was an important starting point. The synset model is sufficiently simple to provide an intrinsic correlation between a concept and the correspondent words. Moreover, WORDNET covers the entire English lexicon and provides an extraordinary large amount of conceptual distinctions. WORDNET is particularly useful from a computational point of view because it was developed for easy access and navigation through its hierarchies. Starting from WORDNET we selected a subset of synsets (named *WORDNET-AFFECT*) suitable to represent affective concepts. We are actually aiming at exploiting the expressivity of the WORDNET model without having to introduce modifications in the original structure. Therefore, we added additional information to the affective synsets without defining new ones. Similarly to the “domain” label we attach to synsets in WORDNET DOMAINS, we assign to a number of WORDNET synsets one or more affective labels (*a-labels*) that contribute to precise the affective meaning. For example, the affective concepts representing emotional state are individuated by synsets marked with the a-label EMOTION. The *concepts* that are *not emotional-affective* (e.g. those representing moods, situations eliciting emotions, or emotional responses) are characterized by other a-labels.

WORDNET-AFFECT was developed in two stages. The first consisted of the identification of the *core synsets*. The second step consisted of the extension of the core with the relations defined in WORDNET.

5.1 The development of the *core* of WORDNET-AFFECT

The first stage of the WORDNET-AFFECT development consists of collecting an initial set of affective words. To this aim, a preliminary resource (named *Affect*) was manually realized.

Affect is a lexical database containing 1,903 terms directly or indirectly referring to mental (e.g. emotional) states. The main part of Affect consists of nouns (539) and adjectives (517). There is a smaller number of verbs (238) and a tiny set of adverbs (15). In order to collect this material, we started from an initial set of psychological adjectives (in particular, affective adjectives). The collection was extended with the help of dictionaries. In a second step, nouns were added through an intuitive correlation with the adjectives. In a similar way, verbs and adverbs were added. For each item a frame was created in order to add lexical and affective information. Lexical information includes the correlation between English and Italian terms, parts of speech (pos), definitions, synonyms and antonyms. The attribute POSR relates terms having different pos but pointing to the same psychological category. For example, the adjective *cheerful* is semantically linked to the name *cheerfulness*, to the verb *cheer up* and to the adverb *cheerfully*. Affective information is a reference to one or more of the three main kinds of theories on emotion representation: discrete theories (based on the concept of cognitive evaluation), basic emotion theories and dimensional theories. According to the work of Ortony et al. (1987), terms are classified in emotional terms, non-emotional affective terms (e.g. *mood*) and non affective mental state terms. Other terms are linked with personality traits, behaviors, mental attitudes, physical or bodily states and feelings (such as *pleasure* or *pain*). Some examples terms and their category are given in Table 3.

Category	*Example Term*
Emotion	anger
Cognitive state	doubt
Personality	competitive
Behaviour	cry
Mental attitude	skepticism
Feeling	pleasure

Table 3: Categories and terms**.**

Discrete emotional information is characterized by an attribute whose value corresponds to one of the 24 emotional categories described by Elliot [1992]. Another attribute allows us to indicate one of the six basic emotions cited by Ekman [1992]. Dimensional emotional information needs two attributes denoting valence (that is, how positive or negative a fixed emotional state is) and arousal (that is the level of emotional excitation).

Part of the information was collected from dictionaries and from scientific documentation on the psychology of emotions; the remaining information was inserted on an intuitive and arbitrary basis. The former kind of data was associated with references to the sources; the latter is the rough material for a subsequently critical review (for example, by psychologists or lexicographers). As an example, here is one of the frames from the database:

```
[name]: anger
[ita]: <rif src=c> rabbia, collera </rif>
      <rif src=wn sense=1> ira, collera, arrabbiatura, rabbia</rif>
      <rif src=wn sense=2> collera, ira, bile, furia, rabbia</rif>
[def]: <rif src=wn sense=1> (Psychology) a strong emotion; a feeling
      that is oriented  toward some real or supposed grievance </rif>
      <rif src=wn sense=2> (Physiology) the state of being angry</rif>
[synonyms]: <rif src=wn sense=1>choler, ire</rif>
      <rif src=wn sense=2>angriness</rif>
      <rif src=mw>  fury, indignation, ire, mad, rage, wrath</rif>
[antonyms]: <rif src=mw>forbearance</rif>
[pos]: n
[posr]: <v>anger</v> <a>angry</a> <r>angrily</r>
[fundamental]: <rif src=d>a</rif>
[elliot]: anger
[valence]: -
[arousal]: 2
[ortony]: emotion
[notes]:
```

5.2 A-labels projections

By mapping the senses of terms in Affect to their respective synsets, the affective core was identified. Then we projected part of the affective information in the Affect database onto the corresponding senses of WORDNET-AFFECT, as value of an affective mark named *a-label* (in the appendix the complete list of a-labels is reported). The information projected was that of the Affect slot *ortony* (used to discriminate between different types of affective concepts). This operation was not complete over all synsets of WORDNET-AFFECT, both because the value of the *ortony* slot was null for some of the Affect items, and because there are synsets manually added besides those individuated in Affect. For this reason, we proceeded to a further manual labeling, in order to assign the a-label value to the whole set of affective synsets.

The opportunity to interface Affect with WORDNET allows us to outline different developments. On one hand it is possible to extend the collection through a search of synonyms and antonyms (performed on each of the terms of Affect that are contained in WORDNET). On the other hand it is useful to compare the affective information of the database with WORDNET hyperonym hierarchy restricted to the Psychology domain [Magnini and Cavaglia 2000], in order to propose enrichment in the structure of this semantic field. Our analysis of WORDNET synsets that contain Affect words suggested that synsets could be used to represent affective concepts. At the level of single affective concepts the characterization as synsets is quite accurate, though additional WORDNET relations, such as those resulting in the ISA hierarchy, are not always appropriate. We have given the name WORDNET-AFFECT to the subset of WORDNET that includes 1,314 synsets representing the senses of the entries in Affect.

	#Nouns	*#Adjectives*	*#Verbs*	*#Adverbs*	*#Total*
#Synsets	535	557	200	22	1314
#Words	1336	1472	592	40	3340

Table 4: Number of affective synsets and words, grouped by part of speech, in WORDNET-AFFECT.

5.3 Exploiting the WORDNET hierarchies

As explained in section 4, in WORDNET a fixed number of lexical (i.e. between words) and semantic (i.e. between synsets) relations are defined. Once we individuated the affective core, we studied if and at what extent, exploiting the WORDNET relations, the affective core of WORDNET-AFFECT could be extended.

For each relation, we examined if it preserves the affective meaning (i.e. if that relation, applied to the synset of WORDNET-AFFECT, generates synsets that yet represent affective concepts). If the resulting synsets are members of WORDNET-AFFECT, the answer is trivially affirmative. But in the case in which the relation generates synsets not included in the database, it should be necessary to proceed to manual checking. However, an exploratory examination allowed us to individuate a list of “reliable” relations (*antonymy*, *similarity*, *derived-from*, *pertains-to*, *attribute*, *also-see*), for which we assumed that the affective meaning is preserved for all items of

WORDNET-AFFECT. Therefore, all synsets obtained by an application of those relations and not yet contained in WORDNET-AFFECT are, de facto, included in it.

For other relations (such as *hyperonymy*, *entailment*, *causes*, *verb-group*) we assumed that the affective meaning is only partially preserved. In that case it is necessary to manually filter the synsets in order to select those genuinely affective.

	#Nouns	#Adjectives	#Verbs	#Adverbs	#Total
similar-to	0	668	0	0	668
antonym	64	106	23	6	199
pertains-to (*direct*)	0	2	0	0	2
pertains-to (*inverse*)	16	0	0	0	16
derived-from (*direct*)	0	0	0	12	12
derived-from (*inverse*)	0	308	0	0	308
also-see	0	148	11	0	159
attribute	38	0	0	0	38
is-value-of	0	30	0	0	30

Table 5: Number of new affective synsets obtained applying WORDNET relations to the synsets of WORDNET-AFFECT.

6. OpenMind-Affect

Once a fixed set of affective concepts has been identified, represented by the synsets of WORDNET-AFFECT, it seems useful to distinguish those directly referring to affective states from those denoting, for example, their causes and consequences. For that it is very important to have a wide resource of common sense expressions. This resource can be exploited as stereotypical knowledge, which allows us to extract relevant information (e.g. events that typically causes specific emotions). With reference to the work of Liu et al. [2003], we have used OpenMind as a source of stereotypical knowledge. OpenMind is a wide common sense knowledge base containing sentences, linguistic patterns and parse trees. Unlike [Liu et al. 2003], sentences of OpenMind were annotated through a word sense disambiguation tool, developed at ITC-irst [Magnini et al. 2002], in order to associate each word with the corresponding WORDNET sense. In this way, it was possible to identify the sentences containing words with an affective meaning and to select an affectively significant subset of OpenMind, which we have called OpenMind-AFFECT. Using only words in Affect, we selected a set of 74,455 sentences in OpenMind. Using the words in WORDNET-AFFECT, we increased the size of this set to 171,657 sentences. This resource is employed as an environment for experimentation about an affective lexicon. In particular, we aim at obtaining the following results:

1. increasing the collection of affective concepts. To this aim, we need to identify, in the sentences of OpenMind-Affect, new words related to well-known ones, in order to obtain new synsets to include in WORDNET-AFFECT.
2. collecting contextual information, such as events that typically cause specific emotions.
3. using contextual information in order to increase the affective knowledge of the lexical items, whenever possible.

In order to extract the contextual knowledge, it is necessary to exploit some linguistic patterns to connect words denoting affective states with contextual words. For example, the pattern X causes Y, where X denotes an event and Y refers to an emotional state, allows us to identify a typical cause of that emotion. The lexical semantics of emotional adjectives [Goy 2000] allows us to deduce some of these structures even if they are not explicitly present in the sentence. For example, the adjective “cheerful” in general may have different ways for denoting the emotional state (stative, manifestative, causative). Nevertheless, if it is included in the noun phrase “cheerful flower”, it assumes the causative reading and implicitly expresses the fact that the flower causes cheerfulness, which allows us to add a potentially new contextual concept to the set of the affective concepts.

7. Future work

At this stage of the work, we put our efforts towards the collection of an initial set of affective concepts. We performed manual tagging in order to have a *core*, then we employed some WORDNET relations in order to extend the core and to propagate the labelling.

Our next task consists of the definition of an appropriate procedure for the a-label annotation of the synsets. We are thinking of evaluating that annotation scheme with a test of reliability and performing an evaluation of the a-label values with an *intertagging agreement* procedure with multiple judges.

We aim to extend the number of affective synsets not only with WORDNET relations but also with machine learning techniques applied to large linguistic corpora. Finally, we want to test WORDNET-AFFECT through an evaluation of the applicative tools in which that resource will be used. For example, considering the affective text sensing tools, WORDNET-AFFECT allows such systems to perform reasoning not only on the "word" level but on the "sense" level (through the synsets linked to each word in the resource). In this manner we think we can obtain an improvement in the performance of the affective understanding from texts. The a-label distinction between *emotion*, *mood*, and *trait* allows user-modeling systems to perform more complex distinctions and so improve the richness of the user model itself. We are thus working in the development of a number of prototypes with which we want to test the applicative potentiality of WORDNET-AFFECT.

8. Conclusions

WORDNET-AFFECT is potentially of wider use, as it allows for some reasoning capability and, as we pointed out, is also connected to relevant common sense knowledge, in textual format. An affective lexicon is *per se* an important resource for many applications, both based on language recognition and on language production. The potential applications in natural language processing are the basis for those in human-computer interaction. In particular, language recognition is employed in user modeling, and language production is necessary for the verbal communication of emotions.

In dealing with texts automatically, emotion-related contents can be retrieved or summarized or seriously classified only if we start from some level of lexical reasoning. If a system is able to perform some reasoning starting from emotion-inducing lexical entries, it may be substantially more sophisticated. In our case it may do so just because of the fact that the common sense thesaurus is linked to synonym sets. For instance we may have knowledge stating that people are afraid of earthquakes. At least some of the simplest statements in the common sense base can have been parsed and yielded semantic relations that involve affect. So whenever earthquakes are a topic, the system could also know they produce fear and therefore classify or summarize the text with this additional dimension. Along the same view all tasks that involve disambiguation can substantially exploit the resource.

The multilingual framework of WORDNET DOMAINS developed at IRST also accommodates WORDNET-AFFECT and this is an important resource for automatic translation, and can yield appropriate ways to overcome language gaps systematically.

On the basis of the lexical information, a system can understand explicitly stated information about the emotional state of the user or of someone else and so the system can use emotion expressions appropriately.

One class of potential applications concerns some (even limited) form of reasoning in a dialog system. For instance a system may be able to advise a user interested in choosing a movie taking into account the fact that he said he does not want to see a *scary* movie. In general the system may want to understand statements regarding the affective state of the user, a simple case is automatic analysis of a questionnaire. Whatever the indicators are (there are experiments with physiologically-based feedback), a system may adapt its expressions to the user.

If we consider system output, there are many aspects with potential for application of the affect lexical base. Multimodal interfaces are a privileged case: for instance a life-like character may appear as the agent delivering the message and the system may coordinate appropriate facial expression when emotional expressions are uttered; similarly the synthetic voice may reflect the emotional valence of a specific fragment of the message. Another case that can be interesting concerns Kinetic Types, where types move producing an emotional effect, normally in agreement with the contents.

Still other possibilities concern the use of persuasive expressions. A system may have to make reference to emotional concepts in order to persuade the audience to perform an action (see Guerini, Stock and Zancanaro 2003).

We see similar potential for games, in the future an essential motivating context for learning environments. Emotional concepts are at the basis of many games and if the interaction between player and system is going to become more natural and complex, they are likely to be expressed linguistically. In a system that helps group activity or in other cases of human-human interaction, it may be important that the system provides different messages to different participants, with appropriately different emotional expressions, even if conveying the same semantic contents.

Finally, we are particularly interested in automatic humour production.

Computational humour can have an important role in future interfaces. Humour plays on the beliefs and expectations of the hearer. By infringing on them, it causes surprise and then hilarity. Humour also encourages creativity. The change of perspective caused by humorous situations induces new ways of interpreting the same event. A typical use of humour can be found in advertisement. We aim at building semiautomatic (and later on fully automatic) systems for helping obtaining novel verbal humour expressions, for instance, in advertising applications. For that purpose we intend to use WORDNET-AFFECT with its connected common sense emotional expressions in a manner that is creative and disrupts conventions.

9. Acknowledgments

This work was realized as part of the TICCA project, funded by the Autonomus Province of Trento and the National Council for Research, and of the HUMAINE network of excellence, funded by the European Commission.

10. Appendix: A complete list of *a-labels* used in WORDNET-AFFECT

A-Label	*Examples*
EMOTION	noun "anger#1", verb "fear#1"
MOOD	noun "animosity#1", adjective "amiable#1"
TRAIT	noun "aggressiveness#1", adjective "competitive#1"
COGNITIVE STATE	noun "confusion#2", adjective "dazed#2"
PHYSICAL STATE	noun "illness#1", adjective "all_in#1"
EDONIC SIGNAL	noun "hurt#3", noun "suffering#4"
EMOTION-ELICITING SITUATION	noun "awkwardness#3", adjective "out_of_danger#1"
EMOTIONAL RESPONSE	noun "cold_sweat#1", verb "tremble#2"
BEHAVIOUR	noun "offense#1", adjective "inhibited#1"
ATTITUDE	noun "intolerance#1", noun "defensive#1"
SENSATION	noun "coldness#1", verb "feel#3"

Table 6: The *a-labels.*

EMOTION synsets are the only pure emotional synset: they refer directly to emotional states/processes.

From theories of emotions we have that moods have to be distinguished from emotions. Therefore, MOOD synsets are affective, but not emotional.

If personality traits influence attitude to have affective reactions to events, then the "TRAIT" synsets have an indirect reference to the affective states.

Affective states are often confused with cognitive ones because they have a mutual strict relation. COGNITIVE STATE synsets represent cognitive states that are often related to affect but form a distinct category.

Similar considerations have to be made for PHYSICAL STATE synsets.

Edonic signals (pleasure and pain) are a necessary component of the affective state but they are not emotions themselves. We introduced the a-label EDONIC SIGNAL in order to cope with these concepts.

Emotional states have causes and consequences. The causes are named ELICITING-EMOTION SITUATIONS and may be events, behaviors etc. (e.g. a "dangerous situation" may cause fear). The consequences of emotions are named "emotional responses", and may consist of behaviors, somatic changes, facial expressions etc. The EMOTIONAL RESPONSE synsets represent that set of responses.

BEHAVIOUR synsets represent behaviors that are either the cause or effect of affective states.

ATTITUDE synsets refer to "attitudes", which are complex mental states involving beliefs, feelings, values, and dispositions to act in certain ways.

A "sensation" is an unelaborated elementary awareness of stimulation. SENSATION synsets represent those kinds of concepts.

11. References

[Artale et al. 1997] Artale, A, Magnini, B., Strapparava, C. (1997). WordNet for Italian and Its Use for Lexical Discrimination. AI*IA97: Advances in Artificial Intelligence. Springer Verlag.

[Briscoe 1991] Briscoe, T. (1991). Lexical Issues in Natural Language Processing, in Klein E. and Veltman F. (eds.): *Esprit Symposium on Natural Language and Speech,* Berlin, Springer-Verlag.

[Calzolari 1992] Calzolari N. (1992). Acquiring and Representing Semantic Information in a Lexical Knowledge Base, in Pustejovsky J. and Bergler S. (eds.) *Lexical Semantics and Knowledge Representation* , Springer-Verlag, Berkeley.

[De Rosis and Grasso 1999]: De Rosis, F. and Grasso, F. (1999). Affective Text Generation. in Proc. of *International Workshop of Affect in Interactions* (IWAI99). Siena, Italy.

[D'Urso and Trentin 1998]: D'Urso, V., Trentin, R. (1998). *Introduzione alla psicologia delle emozioni*. Ed. Laterza.

[Ekman 1992]: Ekman, P. (1992). An argument for basic emotion. *Cognition and Emotion*, 6, 169-200.

[Elliot 1992]: Elliot, C.D. (1992). *The Affective Reasoner: a process model of emotions in a multi-agent system*. Northwestern University doctoral thesis. Evanston, Illinois.

[Fellbaum 1998]: Fellbaum, C., Ed. (1998). *Wordnet: an electronic lexical database*. MIT Press.

[Fussell 2003]: Fussell, S. R., Ed. (2003). *The verbal communication of emotion: Interdisciplinary perspectives*. Mahwah, NJ:Lawrence Erlbaum Associates.

[Goy 2000]: Goy, A. (2000). Lexical Semantics of Emotional Adjectives. In Feist, S., Fix, S., Hay, J., and Moore, J. (eds): *Linguistics in Cognitive Science: Proceedings of Student Conference in Linguistics 10*, MIT Working Papers in Linguistics 37, MIT Press.

[Guerini, Stock and Zancanaro. 2003]: Guerini, M., Stock, O., and Zancanaro, M. (2003). Towards Intelligent Persuasive Interfaces. *Proc. of the IJCAI2003 Workshop on Natural Argumentation*. Acapulco, Mexico.

[Liu et al. 2003]: Liu, H., Lieberman, H., Selker, T.(2003). A Model of Textual Affect Sensing using Real-World Knowledge. *Proceedings of the Seventh International Conference on Intelligent User Interfaces* (IUI 2003),pp. 125-132. Miami, Florida.

[Magnini and Cavaglia 2000]:Magnini, B. and Cavaglia, G. (2000). Integrating Subject Field Codes into WordNet. In: Gavrilidou, M., Crayannis, G., Markantonatu, S., Piperidis, S., Stainhaouer, G. (eds.): *Proceedings of LREC-2000, Second International Conference on Language Resources and Evaluation Theoretical Aspects of Computer Software*. Athens, Greece, 1413-1418.

[Magnini and Strapparava 1997] Magnini, B., Strapparava, C. (1997). Costruzione di una base di conoscenza lessicale per litaliano basata su WORDNET. In *Proceedings of the XXVII Congresso Internazionale di Studi della Società di Linguistica Italiana Linguaggio e Cognizione*, Roma, Bulzoni.

[Magnini et al. 2001] Magnini B, Strapparava C., Pezzulo G., and Gliozzo A. (2001). Using domain information for word sense disambiguation. In *Proc. of SENSEVAL-2*.

[Magnini et al. 2002]: Magnini, B., Strapparava, C., Pezzulo, G., and Gliozzo, A. (2002). The Role of Domain Information in Word Sense Disambiguation. *Natural Language Engineering*, 8(4):359-373.

[Miller 1990] Miller, G. (1990) An on-line lexical database. *International Journal of Lexicography*, 13(4):235-312.

[Nowlis and Nowlis 1956]: Nowlis, V., Nowlis, H.H. (1956). The description and analysis of mood. *Annals of the New York Academy of Sciences*, 65, 345-355.

[Ortony and Clore 1981]: Ortony, A. and Clore, G.L. (1981). Disentangling the Affective Lexicon. In *Proceedings of the Third Annual Conference of the Cognitive Science Society*, Berkeley.

[Ortony et al. 1987]: Ortony, A., Clore G. L., and Foss, M. A. (1987). The psychological foundations of the affective lexicon. *Journal of Personality and Social Psychology*, 53, 751-766.

[Ortony et al. 1988]: Ortony, A., Clore, G.L., Collins, A. (1988). The cognitive structure of emotions. Cambridge University Press, New York.

[Picard 1997]: Picard, Rosalind (1997). Affective Computing. MIT Press.

[Picard 2000]: Picard, R.W. (2000). *Toward computers that recognize and respond to user emotion*. IBM Systems Journal, 39(3&4):705-719.

[Singh et al. 2002]: Singh, P., Lin, T., Mueller, E. T., Lim, G., Perkins, T., and Zhu, W. L. (2002). Commonsense: Knowledge acquisition from the general public. In *Proceedings of the First International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems*, Heidelberg. Springer-Verlag.

[Stock and Strapparava 2003]: Stock, O. and Strapparava, C. (2003). Getting Serious about the Development of Computational Humor. In *Proceedings of International Joint Conference of Artificial Intelligence* (IJCAI-03), Acapulco, Mexico.

[Young and Hamer 1987]: Young, F.W. & Hamer, R.M. (1987) Multidimensional Scaling: History, Theory and Applications Erlbaum, New York.

[Watson and Tellegen 1985]: Watson, D., Tellegen, A. (1985). Toward a consensual structure of mood. Psychological Bullettin, 98, 219-235.

[Weigand 1998]: Weigand, Edda. (1998). Contrastive Lexical Semantics. John Benjamin Publishing.

PsychNology Journal, 2004
Volume 2, Number 1, 84 - 98

Finding Geometrical Associations Between Meaningful Objects in the Web: A Geostatistical Approach

Francesco Di Nocera[♣], Corinne Capponi, Fabio Ferlazzo

Cognitive Ergonomics Laboratory, Department of Psychology
University of Rome “La Sapienza”, Italy

ABSTRACT

The study reported in this paper was aimed at investigating the existence of schemata specifically involved in the cognitive organization of a web page. Particularly, the hypothesis was that the location of some web objects (namely, links to specific contents) might be expected by the users at specific spatial locations. Using a method providing geometrical information concerning the organization of web contents, we found that user’s expectations could be linked to the activity of low- and high-level schemata allowing performance optimization. Potential benefits of the Cognitive GeoConcept procedure for supporting information architects’ decisions are discussed.

Keywords: *schemata, spatial representations, spatial point patterns, page layout, usability*

Received 19 January 2004; received in revised form 1 April 2004; accepted 2 April 2004.

1. Introduction

How people manage to operate in a complex world is a long lasting problem in Psychology. Human skilled performance has been described in several different ways, and cognitive models have been derived. Among the others, research in Cognitive Ergonomics strongly contributed to this theoretical mission, investigating human performance in real-world tasks from text selection (Card, English, and Burr, 1978) to driving (Hale, Quist, and Stoop, 1988; Summala, 1988), just to name a few.

The general perspective we will endorse throughout this paper is that interaction with technological artifacts happens by means of representations or schemata (see Norman and Shallice, 1986 for an account on this construct) allowing the optimization of our behaviors. This is a very accredited perspective, albeit definitions of schema and optimization may differ among scholars. For example, Tversky and Kahneman (1974) proposed the idea of heuristics that are shortcuts supporting human decision-making

♣ Corresponding Author:
Francesco Di Nocera, Ph.D.
Cognitive Ergonomics Laboratory, Department of Psychology
University of Rome "La Sapienza", via dei Marsi 78, 00185 Rome, Italy
Tel.: +39(06)49917663 - Fax: +39(06)4451667
E-mail: francesco.dinocera@uniroma1.it

(see Kahneman, 2003 for a recent account on heuristics), whereas others have emphasized the role of physical, environmental, and social contexts (see Suchman, 1987 for a general perspective on situatedness). All these contributions have in common the idea that the human cognitive system selects the most effective strategies for interacting with other people, objects, and -more generally- events.

The present research paper, however, moves from considerations mainly derived from the work of John R. Anderson (Anderson, 1991; but see also Anderson, 2002 for a recent review), who proposed the existence of an optimization criterion within his rational analysis. Recently, our laboratory applied this very rationale to the field of traffic psychology (Couyoumdjian, Di Nocera, and Ferlazzo, 2002), showing the existence of an optimization criterion for speed selection aimed at minimizing mental workload. This mechanism was related to the activity of expertise-based schemata favoring expert drivers, suggesting that (automatic) workload control is a learned strategy.

Since the abstraction of the approach described above, interaction with web sites may be conceptualized as governed by schemata as well. Those schemata would be aimed at the optimization of user's behaviors during navigation tasks. Some of them may be involved in the way people evaluate web sites (see Di Nocera, Ferlazzo, and Renzi, 1999; in press, for a model consistent with these ideas), whereas others may underlie the types of actions that are executable within a web site (namely, searching, browsing, pointing, writing, and the like). Furthermore, one additional class of schemata may be specifically involved in the way people look for information either within the entire site or the web page. Such hypothetical schemata might refer to a prototypical organization of the information, and would contain rules and specifications for the location of 1) objects within the page layout, and 2) contents within the site structure.

It is commonly accepted that schemata would be hierarchically organized: from general, expertise-dependent schemata to lower-level schemata that are strictly intertwined with structural and functional constraints in the cognitive system. For instance, page scanning often occurs consistently with reading direction, which is culturally-dependent, but not expertise-dependent. Tversky, Kugelmass, and Winter (1991) reported that culture also affects graph reading on the x-axis. People show better comprehension of graphs organized according to the reading direction of their printed language. On the contrary, culture seems to have no effect on y-axis graph reading: everybody shows a preference for graph organizing high values at the top and low values at the bottom of the axis. Of course, the reading direction schema is not located at the very bottom of the hierarchy. Yet, to our aim, this is still a good level of specification.

2. Spatial representations for web sites?

A leitmotiv of any Human-Computer Interaction handbook is that effective design cannot leave out knowledge about the mental models of the user. Any cognitive-based perspective agrees with the idea that if system appearance (the visible structure of the system which acts as a filter between the user and the designer) is not consistent with design (the outcome of the designer's conceptual model), the user will likely experience disappointment, frustration, and stress. Understanding how individuals find objects in a web page may favor a design reflecting the type of organization the user expects. This would make the sites more accessible, easy to browse, and satisfactory. Yet, the idea that the way people relate to objects deployed in space has a critical role in interface design is not new. Several authors (Rosenfeld and Morville, 1998; Wurman and Bradford, 1996) showed that interaction with hypertexts is strongly affected by users' spatial knowledge. Card sorting, for example, is a practice that has been proven to be very useful for organizing informative units into a hierarchical structure, particularly when the amount of information to be delivered is abundant (Maiden and Hare, 1998). However, the elements arranged into a single web page should be also deployed in a way suited for optimizing navigation. Unfortunately, research on this issue is scarce. The only empirical study on expected location for web objects is that of Michael Bernard (Bernard, 2001). Regrettably, this research was affected by serious methodological problems. For example, users did not interact with a computer, but with a depiction of a browser window, on which they had to arrange pictures of links and banners, according to positions defined *a priori* by 8 x 7 grid squares. Most importantly, subjects performed the task only once per web object (twice for advertisement banners). However, we would like to clarify that our considerations about the shortcomings of Bernard's study are only limited to the experimental rigor and by no means to its applied potential. Bernard (2001) used a sort of "paper prototyping" technique (Rettig, 1994), which is indeed quite useful and popular among interaction designers.

The present study is based on a less explicit method, eliciting users' responses to verbal labels indicating the to-be-positioned web objects on a large number of trials. We called this method "Cognitive GeoConcept", for it is aimed at finding geometrical associations between meaningful objects (links to other pages or functions).

According to what discussed above, two different results can be expected from this study:

1. "low-level" schemata (spatially and semantically based) would be involved in the process - both experienced and inexperienced users should show the same pattern of arrangement;
2. "high-level" schemata (mainly based on navigation experience) would be involved in the process - only experienced users should show an interpretable pattern of arrangement.

Of course, motivational and emotional factors may also play a role, as reported in the recent literature on "affective computing" (Picard, 1997). However, they will not be taken into account in the present research paper, as it is primarily aimed at studying spatial schemata.

3. Method

Participants. Twenty-three students (14 females) volunteered in this experiment. Their mean age was 25.2 years. Thirteen subjects reported to use Internet every day, and were classified as experts. Subjects classified as novices reported to navigate few times per week (5 users) or month (5 users). All users reported being right-handed, with normal or corrected to normal vision, and were naïve as to the purpose of the experiment.

Stimuli. Fourteen words indicating links to resources often found in web sites (about us, buy, catalog, check your e-mail, contact us, help, home, jobs, news, play & win, register, resources, restricted area, search) were used as stimuli.

Procedure. Participants sat in front of a 17" computer monitor and received detailed information about the procedure and the meaning of the labels used. They had to respond as quickly as possible to the stimuli by clicking on the area of the (blank) screen where they would expect to find the corresponding link in an imaginary web page. Stimuli were presented centrally, white on black, for 200 ms. On any trial the mouse pointer returned to the center of the screen. Particularly, the procedure can be outlined as follows (see figure 1):

1. a string indicating a link (i.e. “buy”) was centrally administered on the screen;
2. subjects clicked on the portion of the screen where they would expect to find that link;
3. the pointer returned to the center of the screen;
4. another stimulus (i.e. “home”) was presented and a new click was required.

Fifty repetitions of each stimulus were randomly administered to the subjects, making the number of trials very large (700 in the present experiment).

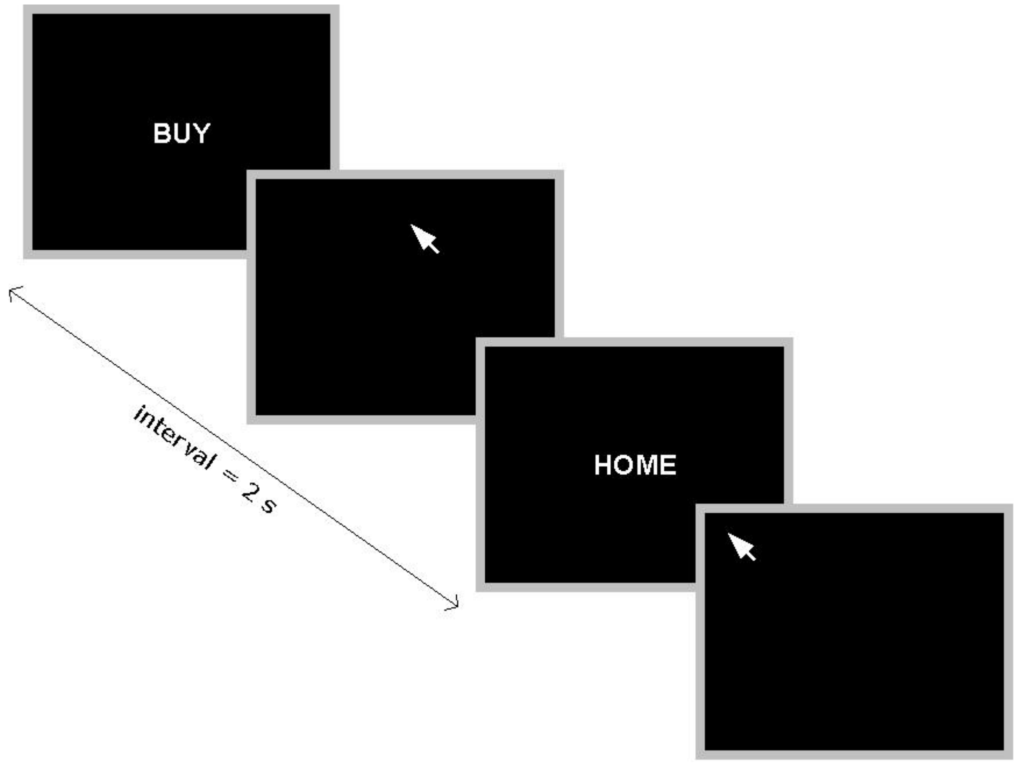


Fig. 1: A graphical representation of the Cognitive GeoConcept procedure.

5. Analyses

Artifacts rejection. Prior to analyze the data, we visually inspected the patterns generated by each label, and found that within the first ten trials the clicks were highly dispersed. However, that was not surprising, because point patterns are known to vary along time. That happens for several reasons, and in our case one of the most important is memory. Indeed, albeit subjects were usually consistent with their initial response pattern, and spontaneously clicked around the same location where they initially “placed” a link, changes in their choice could happen. Another factor determining pattern variability is practice. Our experimental setup provided very fast stimuli administration and required very fast responding. Subjects needed some time to familiarize with the procedure and to become more confident with the task. In both cases, changes could have affected the overall result, because we averaged clicks coordinates. For this reason we run several analyses using time as factor. Particularly, for each stimulus, we divided trials in five blocks of ten trials each. Repeated measures ANOVA designs by Block (1^{st} vs. 2^{nd} vs. 3^{rd} vs. 4^{th} vs. 5^{th}) were run for each stimulus using reaction times as measure. Analysis for “catalog”, “help”, “home”, “buy”, “about us”, “resources” showed no significant differences between reaction times in the different blocks ($p>.05$). Significant effects were found for “register” ($F_{4,84}=2.51$, $p<.05$), “check your e-mail” ($F_{4,88}=2{,}69$, $p<.05$); “news” ($F_{4,88}=7.83$, $p<.001$), “jobs” ($F_{4,84}=4{,}27$, $p<.01$), “play & win” ($F_{4,80}=3{,}57$, $p=.01$), and “contact us” ($F_{4,84}=3.11$, $p<.05$). In all cases significant differences were found between the first block and the others. Accordingly, only data from blocks 2 to 4 for all the stimuli were used in the successive analyses.

Conventional analyses. Coordinates were analyzed using Cluster Analysis (Ward’s method). Input distance matrices (for Experts and Novices) were computed using average point-to-point Euclidean distances. Positioning responses were further examined using quadrat counts. A 4 x 4 grid was used to divide the area in 16 quadrats (figure 2). Angular transformations of the proportion of clicks within the quadrats were then analyzed using a mixed ANOVA design Expertise (Experts *vs*. Novices) x Link (contrasting the 14 different links) x Row (1^{st} *vs*. 2^{nd} *vs*. 3^{rd} *vs*. 4^{th}) x Column (1^{st} *vs*. 2^{nd} *vs*. 3^{rd} *vs*. 4^{th}).

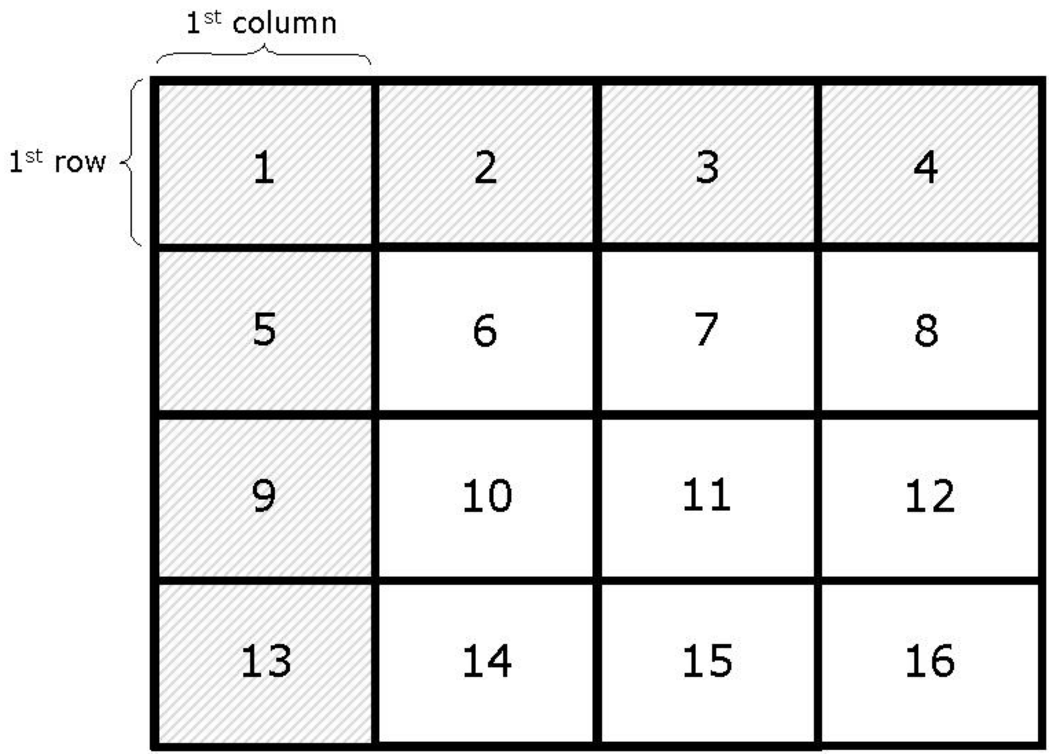


Fig. 2: The imaginary 4 x 4 grid used to partition the screen in 16 quadrats.

Spatial analyses. The spatial equivalent of uniformly and independently distributed random variables is the Complete Spatial Randomness (CSR). Any point pattern distribution analysis needs some sort of comparison with this distribution. In the present study we used this rationale to investigate the distribution of the users' clicks on the screen.

The spatial index we used was the Nearest Neighbor Index (Clark and Evans, 1954), which is one of the most widely used distance statistics. Computing it is very easy, and many other distance statistics are founded on it. As a first step, the nearest neighbor distance or d(NN) should be computed as follows:

$$\mathbf{d(NN)} = \sum_{i=1}^{N} \left[\min \frac{(d_{ij})}{N} \right]$$

where $\min(d_{ij})$ is the distance between each point and the point nearest to it, and N is the number of points in the distribution.

This index is nothing more than the average of the minimum distances. The second step is to compute the mean random distance or d(ran), that is the d(NN) one would expect if the distribution were random.

$$\mathbf{d(ran)} = \mathbf{0.5}\sqrt{\frac{\mathbf{A}}{\mathbf{N}}}$$

where A is the area of the region (the measurement unit of the index is related to the one used here), and N is the number of points.

The final step is the actual computation of the Nearest Neighbor Index as follows:

$$\mathbf{NNI} = \frac{\mathbf{d(NN)}}{\mathbf{d(ran)}}$$

This ratio is equal to 1 when the distribution is random. Values lower than 1 suggest grouping, whereas values higher than 1 suggest regularity (i.e. the point pattern is dispersed in a non-random way).

It is also possible to compute the statistical significance for this index (as we have done in this study). Indeed, Complete Spatial Randomness (CSR) hypothesis was tested separately for Experts' and Novices' clicks distributions using the Nearest Neighbor Index (NNI). However, here we will not enter into the details of statistical inference for spatial data: the interested reader may refer to Haining (2003) for an exhaustive account.

6. Results

CSR test showed that the two point distributions were not random. Experts' NNI was 4.02 ($Z=77.99$, $p<0.01$), whereas Novices NNI was 3.28 ($Z=51.67$, $p<0.01$). Both results indicated regularity. Cluster Analysis showed different patterns for the two groups. Experts showed five clusters (figure 3) named *user input* (register, check your e-mail), *user commitment* (play & win, buy, contact us), *company info* (news, search, about us, home), *corporate identity* (product catalog, jobs), and *access to resources* (resources, restricted area). One link (help) only combined with the others at larger distance. Novices showed six clusters (figure 4): *user input* (buy, search, play & win), *access to resources* (resources, register), *corporate identity* (catalog, jobs), *general* *functions* (home, help), and two uninterpretable clusters. One link (check your e-mail) was separated from the other groups.

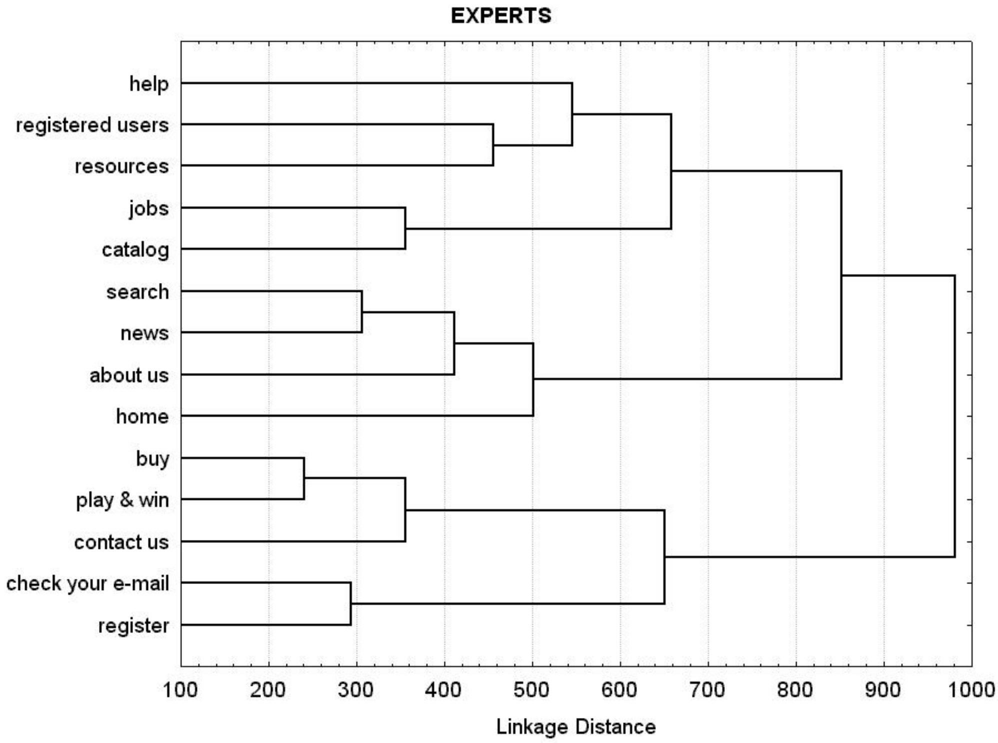


Fig. 3: Horizontal tree diagram summarizing the clustering of links for the experts. Note how easy can be the naming of the clusters in this case.

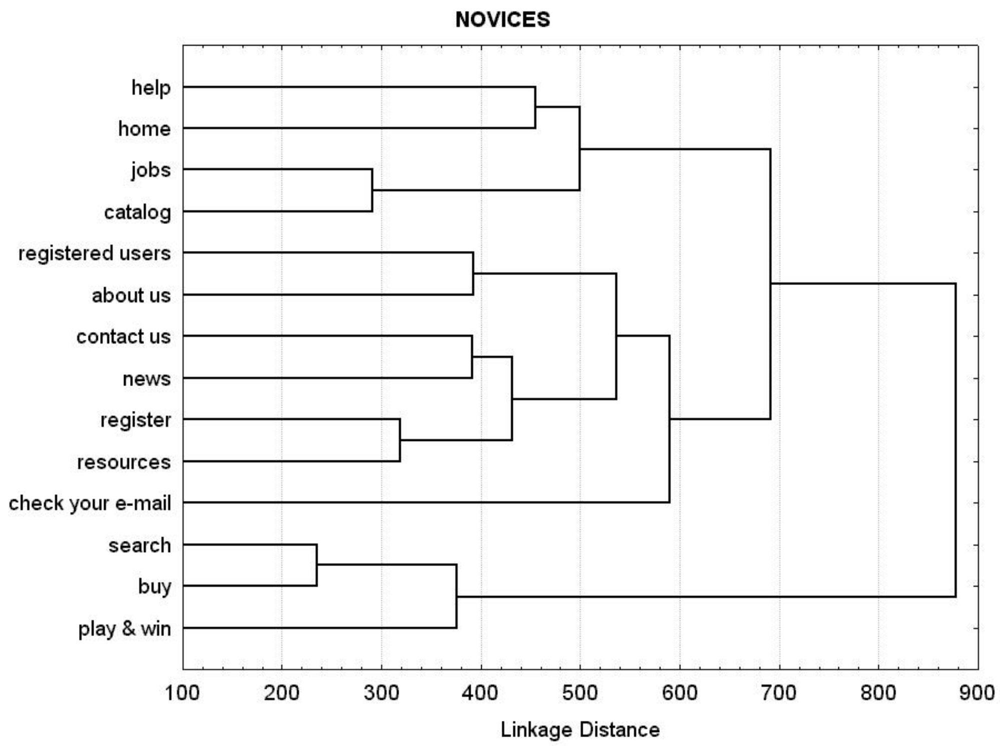


Fig. 4: Horizontal tree diagram summarizing the clustering of links for the novices. Note how difficult can be the naming of the clusters in this case.

The Analysis of Variance showed a significant Expertise x Link x Rows x Columns third order interaction ($F_{117,2457}$=1.63, p<0.01). Thus separate factorial analyses of variance were run for each link. A “Row” main effect was found for the stimuli “check your email”, “registered users”, and “jobs” ($F_{3,63}$=2.80, p<0.05; $F_{3,63}$=2.70, p<0.05; $F_{3,63}$=3.67, p<0.05, respectively). Duncan testing showed that the individuals’ preference for locating those stimuli significantly increased downward. Furthermore both “registered users” and “jobs” showed a main effect of “Column” ($F_{3,63}$=4.31, p<0.01; $F_{3,63}$=3.04, p<0.05, respectively). Duncan testing showed that subjects located both stimuli within the farthest (left and right) columns significantly more often than within the two central columns. Rows by Columns interactions were showed on the proportion of clicks to the stimuli “search” ($F_{9,189}$=2.17; p<0.05), “news” ($F_{9,189}$=2.63; p<0.01), “about us” ($F_{9,189}$=2.50; p=0.01), and “home” ($F_{9,189}$=9.91; p<0.01). High variability of clicks to “search” did not allow any further interpretation of this result. On the contrary, Duncan testing showed that “news” was located in the quadrat defined by the first row and the second column significantly more often than in all the other positions, whereas “about us” was located more often within the first column (with a proportion of clicks increasing from the lower to the upper part of the screen). Finally, "home" was more often assigned to the uppermost-leftmost corner. An Expertise by Row interaction was found for the stimulus "home" ($F_{3,63}$=4.84, p<.01). Duncan testing showed that it was due to the novices locating this link also in the lowest part of the screen. Three links (buy, help, and resources) showed an Expertise by Row by Column interaction ($F_{9,189}$=2.49; p<0.01, $F_{9,189}$=7.32; p<0.01 and $F_{9,189}$=2.32; p<0.05, respectively). Particularly, "buy" was mostly assigned to the lowermost-rightmost corner by experienced users, whereas novices preferred the uppermost-rightmost area, "help" was assigned to the uppermost-rightmost corner only by experts, and "resources" was assigned to the upper part of the screen by expert and novices with opposite patterns (leftward for experts *vs*. rightward for novices).

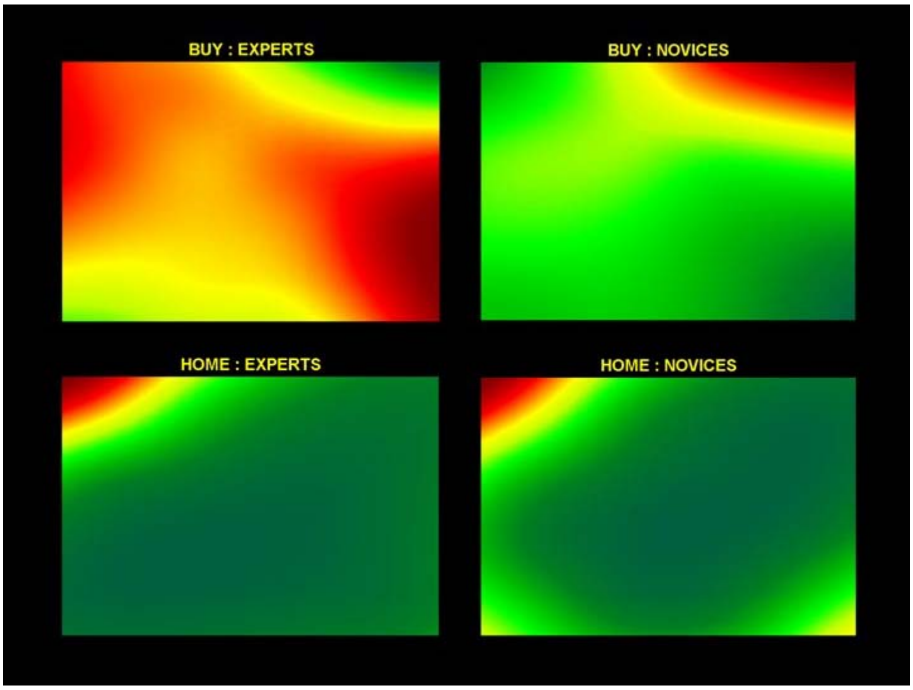


Fig. 5: Spline interpolations for dispersed (buy) vs. concentrated (home) patterns, separately for experts and novices. Dark red indicates high clicking areas, whereas dark green indicates low clicking areas.

7. Discussion

According to Tversky (2003), the invented space of graphics consists of elements (icons, for example) and the spatial relations among them (for example, distance and direction), both conveying meaning. They represent natural correspondences that "reduce load on working memory and cognitive operations" (p. 77). The present study was aimed at investigating the existence of spatial schemata specifically involved in the cognitive organization of web pages, which are invented spaces as well. Our hypothesis was that the location of some links could be expected at specific spatial locations. Expectations would be due to the activity of schemata whose aim is to optimize user's performance. Two possible patterns of results could have been expected, according to the type of schemata involved in the process: either spatially- and semantically-based schemata or schemata based on navigation experience.

Cluster analysis results partially supported the first prediction, as some important clusters were matched for the two groups. However, experts' were the only really interpretable clusters, whereas novices' showed at least two uninterpretable clusters. We are aware that one of the most important issues affecting users' performance was the nature of the stimuli we used. Indeed, our study was general in its scope, and stimuli represented links to functions and resources available in different typologies of websites. Using links from one type of website may provide much clearer clusters.

Analyses performed on single links provided information extending that obtained from cluster analysis. Expected positions matched the most common among websites (i.e. home, help, etc.), supporting the idea that strategies also affect users' performance. Our results are thus in contrast with those reported by Bernard (2001) who found no effect of expertise on spatial organization. Likely, such a discrepancy might be due to the methodological differences between the two studies.

Overall, the theoretical scope of this study was partially fulfilled: "natural" deployment of web objects seems to occur, albeit this is not always the case. Again, some objects showed a common location for experts and novices, suggesting a semantically-based organization. However, we do not exclude that this may be due to subject's previous knowledge about web sites. In fact, one limit of the studies comparing experts' and novices' performance is that novices are never completely fasting from the conventions used in interface design. It is actually hard, nowadays, to find people who never navigated a web site. Thus, we carefully consider the differences only in the degree of knowledge the two groups have.

Natural positioning is somewhat related to the "affordance issue" in interface design. Norman (1999) argued that objects deployed in an interface represent cultural constraints (conventions shared by a cultural group), and they do not provide real affordances (Gibson, 1977; 1979). Albeit we do not completely agree with this account, this single study cannot rule out this possibility, nor can it completely support the idea of "natural" locations for web objects.

Still, the usefulness of the procedure as a technique for supporting information architects' decisions has been attested. In the future, guidelines may be derived from the Cognitive GeoConcept procedure, and may be used to gather clear indications for design.

8. Conclusions

The results of the present study are very encouraging, although more research is needed, as well as replication studies using wider samples and different stimuli. Also, other indexes should be used to assess the validity of the procedure. Indeed, clicks may be affected by processes such as memory and decision-making. Hence, there is a need for more stringent measures, possibly related to early processing. Psychophysiological measures may do, but they are difficult to implement and have some shortcomings. For example, Ward and Marsden (2003) have attempted the use of psychophysiological measures for monitoring physiological activity during the navigation of ill- and well-designed sites. However, different mental events can produce near identical physical responses, representing one of the most important shortcomings. Eye-movements, instead, may represent a more convenient measure. Indeed, collecting unintentional gazes towards particular screen areas during the Cognitive GeoConcept task is technically feasible, and preliminary data we collected at our lab showed consistent patterns between fixations and clicks (consistently with findings reported by Chen, Anderson, and Sohn, 2003). However, this coherence was showed only by novices, suggesting the existence of motor schemata, partially independent from visual processing, in expert users.

Of course, the present study leaves unanswered many questions about the nature of the schemata discussed above. However, it may be useful in the future to make use of the knowledge gained in this domain to test the efficacy of specific page layouts. For instance, one might evaluate the performance of users interacting with websites organizing space according to the groupings found. Also, extending this testing procedure to specific groups of users (i.e. juniors *vs*. seniors) would be useful. This may eventually lead to improve our comprehension of the processes involved in human-technology interaction, and to design "objects" that wait for users' inputs … there where the users expect them to be waiting for.

9. Acknowledgements

We are very grateful to Leandro Agrò and Francesco Maringelli for their useful comments to an earlier version of this paper.

10. References

Anderson, J.R. (1991). The adaptive nature of human categorization. *Psychological Review*, 98, 409-429.

Anderson, J.R. (2003). Is human cognition adaptive? In T.A. Polk (Ed), *Cognitive modeling* (pp. 1193-1228). Cambridge: MIT Press.

Bernard, M.L. (2001). Developing schemas for the location of common web objects. *Proceedings of the Human Factors and Ergonomics Society 45th Annual Meeting*, 1, 1161-1165.

Card, S.K., English, W.K., & Burr, B.J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics*, 21, 601-613.

Chen, M., Anderson, J.R., & Sohn, M. (2003). Eye-hand coordination during Web browsing. In J. Ratner (Ed.), *Human factors and Web development* (2nd ed.) (pp. 207-224). Mahwah: Lawrence Erlbaum Associates.

Clark, P.J., & Evans, F.C. (1954). Distance to nearest neighbor as a measure of spatial relationships in populations. *Ecology*, 35, 445-453.

Couyoumdjian, A., Di Nocera, F., & Ferlazzo, F. (2002). Spontaneous speed: theoretical and applied considerations. In D. de Waard, K.A. Brookhuis, J. Moraal, & A. Toffetti (Eds.), *Human Factors in Transportation, Communication, Health, and the Workplace* (pp. 175-188). Maastricht: Shaker Publishing.

Di Nocera, F., Ferlazzo, F., & Renzi, P. (1999). Us.E. 1.0: costruzione e validazione di uno strumento in lingua italiana per valutare l'usabilità dei siti Internet. In M.F. Costabile, & F. Paternò (Eds.), *Proceedings of HCITALY '99. Report CNUCE-B4-1999-003*. Pisa: CNUCE.

Di Nocera, F., Ferlazzo, F., & Renzi, P. (in press). L'usabilità a quattro dimensioni. *Ricerche di Psicologia*.

Gibson, J.J. (1977). The theory of affordances. In R. E. Shaw & J. Bransford (Eds.), *Perceiving, Acting, and Knowing*. Hillsdale: Lawrence Erlbaum Associates.

Gibson, J.J. (1979). *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin.

Haining, R. (2003). *Spatial Data Analysis: Theory and Practice*. Cambridge: Cambridge University Press.

Hale, A.R., Quist, B.W., & Stoop, J. (1988). Errors in routine driving tasks: A model and proposed analysis technique. *Ergonomics*, 31(4), 631-641.

Kahneman, D. (2003). A perspective on judgment and choice: Mapping bounded rationality. *American Psychologist*, 58(9), 697-720.

Maiden, N.A.M., & Hare, M. (1998). Problem domain categories in requirements engineering. *International Journal of Human-Computer Studies*, 49(3), 281-304.

Norman, D. A., & Shallice, T. (1986). Attention to action: willed and automatic control of behaviour. In R.J. Davidson, G.E. Schwartz, & D. Shapiro (Eds.). *Consciousness and self-regulation, Vol. 4* (pp. 1-18). New York: Plenum Press.

Norman, D.A. (1999). Affordance, conventions, and design. *Interactions*, 3, 38-43.

Picard, R. W. (1997). *Affective Computing*. Cambridge: MIT Press.

Rettig, M. (1994). Prototyping for tiny fingers. *Communications of the ACM*, 37(4), 21-27.

Rosenfeld, L., & Morville, P. (1998). *Information architecture for the World Wide Web*. Sebastapool: O'Reilly & Associates, Inc.

Suchman, L.A. (1987). *Plans and situated action*. New York: Cambridge University Press.

Summala, H. (1988). Risk control is not risk adjustment: the zero-risk theory of driver behaviour and its implications. *Ergonomics*, 31, 491-506.

Tversky, A., & Kahneman, D. (1974). Judgement under uncertainty: heuristics and biases. *Science*, 185, 1124-1131.

Tversky, B. (2003). Structures of mental spaces: How people think about space. *Environment & Behavior*, 35(1), 66-80.

Tversky, B., Kugelmass, S., & Winter, A. (1991) Cross-cultural and developmental trends in graphic productions. *Cognitive Psychology*, 23, 515-557.

Ward, R.D., & Marsden, P.H. (2003). Physiological responses to different web page designs. *International Journal of Human-Computer Studies*, 59(1-2), 199-212.

Wurman, R.S., & Bradford, P. (1996). *Information Architects*. New York: Graphis Press.

PsychNology Journal, 2004
Volume 2, Number 1, 99 - 122

Software Environments for End-User Development and Tailoring

Maria Francesca Costabile♥[1], Daniela Fogli[2],
Giuseppe Fresta[3], Piero Mussio[4], Antonio Piccinno[1]

[1] Dipartimento di Informatica, Università di Bari, Bari, Italy.

[2] Dipartimento di Elettronica per l'Automazione, Università di Brescia, Brescia, Italy
[3] ISTI - CNR, Pisa, Italy.

[4]Dipartimento di Informatica e Comunicazione, Università di Milano, Milano, Italy.

ABSTRACT

In the Information Society, end-users keep increasing very fast in number, as well as in their demand with respect to the activities they would like to perform with computer environments, without being obliged to become computer specialists. There is a strong request of providing end-users with powerful and flexible environments, tailorable to the culture, skills and needs of very diverse end-user population. In this paper, we discuss a framework for End-User Development (EUD) and present our methodology to design software environments that support the activities of a particular class of end-users, called domain-expert users, with the objective of easing the way these users work with computers. Such environments are called Software Shaping Workshops in analogy to artisan workshops, since they provide users with the tools, organized on a bench, that are necessary to accomplish their specific activities by properly shaping software artifacts. The methodology is discussed, outlining its implementation through a web-based prototype.

Keywords: *End-User Development, Visual Interaction, Tailoring, Customization*.

Received 16 January 2004; received in revised form 31 March 2004; accepted 1 April 2004.

1. Introduction

Even if progress has been made in improving the way users can access interactive software systems, some phenomena affecting the life of interactive products still make difficult to develop software systems acceptable in a working environment. In Human-Computer Interaction (HCI), it is often observed that "using the system changes the users, and as they change they will use the system in new ways" (Nielsen, 1994). In turn, the system must evolve to adapt to its new usages; we called this phenomenon

♥ Corresponding Author:
Maria Francesca Costabile
Dipartimento di Informatica,
Università di Bari, Bari, Italy
E-mail: costabile@di.uniba.it

co-evolution of users and systems (Arondi et al., 2002). In (Bourguin et al., 2001), it is observed that these new uses of the system determine the evolution of the users culture and of their models and procedures of task evolution, while the requests from users force the evolution of the whole technology supporting interaction.

Co-evolution stems from two main sources: a) user creativity, i.e. users may devise novel ways to exploit the system in order to satisfy some needs not considered in the specification and design phase; and b) user acquired habits, i.e. users may follow some interaction strategy to which they are (or become) accustomed; this strategy must be facilitated with respect to the initial design.

Co-evolution implies tailoring that, according to (Mørch and Mehandjiev, 2000), is "the activity of modifying an existing computer system in the context of its use, rather than in the development context". This definition emphasizes that users themselves can tailor the system to their necessities. Tailoring stems from a continuous adaptation of a system and is seen as the indirect long-term collaboration between developers and users. Tailoring should be driven by users to exploit the potential benefits of task-oriented and skill-based system adaptations that only users themselves can perform. However, a trade-off to this approach is the variety of developed applications to be maintained by software engineers. Our proposal is also aimed at coping with this problem.

One fundamental challenge for the coming years is to develop environments that allow people without particular background in programming to develop and tailor their own applications, still maintaining the congruence within the different evolved instances of the system. Over the next few years, we will be moving from *easy-to-use* (which has yet to be completely achieved) to *easy-to-develop-and-tailor* interactive software systems.

People who will mainly benefit from this perspective shift are *end-users*, i.e. those persons who, according to Cypher, use a computer application as part of daily life or daily work, but are not interested in computers per se (Cypher, 1993). It is evident that several categories of end-users can be defined, for instance depending on whether the computer system is used for work, for personal use, for pleasure, for overcoming possible disabilities, etc. End-user population is not uniform, but divided in non-mutually exclusive communities characterized by different goals, tasks and activities. Even these communities cannot be considered uniform, because they include people with different cultural, educational, training, and employment background, novice or experienced in the use of the computer, the very young and the elderly, and with different types of (dis)abilities. End-users operate in various interaction contexts and scenarios of use, they want to exploit computer systems to improve their work, but often complain about the difficulties in the use of such systems. The challenge for designers is to develop interactive systems customized to a community, without loosing the generality and the power of computer tools.

Our experience is focused on a particular class of end-users, that we call *domain-expert users* (or *d-experts* for short): this kind of users, such as medical doctors, mechanical engineers, geologists, etc., are experts in a specific domain, not necessarily experts in computer science, and use computer environments to perform their daily tasks. D-experts have the responsibility for possible errors and mistakes, even those generated by wrong or inappropriate use of the software.

This paper describes a method to overcome the trade-off between customization and generality in the case of d-experts.

The ultimate aim is empowering end-users and d-experts to flexibly employ advanced information and communication technologies within the future environments of ambient intelligence. To this aim, the European Community has recently funded EUD-Net, a network of Excellence on End-User Development (EUD).

This paper provides the following contributions to the research on EUD:

1) an analysis of the need of developing software that domain-expert users have;
2) a design methodology to build
    a. software environments tailored to the needs of the considered user community,
    b. tools allowing d-experts to design and develop software environments in collaboration with software engineers and HCI experts.

The proposed design methodology is the evolution of the design strategy described in (Carrara et al., 2002a, Costabile et al. 2002), and allows both end-users and d-experts to perform EUD activities, including tailoring of existing software environments and creation of new software artifacts.

The paper is organized as follows: Section 2 provides insights to the concept of EUD. In Section 3, an analysis of needs of EUD, that domain-expert users have, is reported. Section 4 illustrates the Software Shaping Workshop methodology. Section 5 discusses the Interaction Visual Language used in the design methodology. Section 6 presents an example of the application of the SSW methodology to a real case. Section 7 concludes the paper.

2. End-User Development

New technologies have created the potential to overcome the traditional separation between end-users and software developers. New environments able to seamlessly move from using software to programming (or tailoring) can be designed. Advanced techniques for developing applications can be used by individuals as well as by groups or social communities or organizations.

Some studies say that by 2005, there will be in USA 55 millions of end-users compared to 2.75 millions of professional users (Boehm et al., 2000). End-users population is not uniform, but it includes people with different cultural, educational, training, and employment background, novice and experienced computer users, the very young and the elderly, people with different types of (dis)abilities. Moreover, these users operate in various interaction contexts and scenarios of use and they want to exploit computer systems to improve their work, but often complain about the difficulties in the use of such systems.

Based on the activity performed so far within the EUD-Net network of excellence, the following definition of EUD has been proposed: "*End-User Development is a set of activities or techniques that allow people, who are non-professional software developers, at some point to create or modify a software artifact".* EUD means the active participation of end-users in the software development process. In this perspective, tasks that are traditionally performed by professional software developers are transferred to the users, who need to be specifically supported in performing these tasks. The range of active user participation in the software development process can range from providing information about requirements, use cases and tasks, including participatory design, to end-user programming. Some EUD-oriented techniques have already been adopted by software for the mass market such as the adaptive menus in MS Word™ or some programming-by-example techniques in MS Excel™. However, we are still quite far from their systematic adoption.

EUD is based on the differences among end-users, professional programmers and software engineers. There are differences in training, culture, skill and technical abilities, in the scale of problems to be solved, in the processes, etc.

Within the EUD-Net activity, the following research directions have been identified as fertile for allowing end-users to *craft* software: 1. theoretical and empirical studies of what problems addressed by software engineering transpose to EUD, why and how; 2. studies to identify possibly existing problems that are specific to EUD and are thus not addressed by software engineering; 3. research on methods and tools that would address the previously identified problems in ways that are adequate for end-users: "lightweight methods", tools to support them, and offering appropriate user interfaces taking into account end-users tasks and activities.

Our proposal of designing Visual Interactive Systems organized as environments called Software Shaping Workshops, which will be illustrated in Section 4, is in the direction of point 3 above.

3. User needs of EUD

We often work with end-users that are experts in their field, that need to use computer systems for performing their work tasks, but that are not and do not want to become computer scientists. This has motivated our definition of domain-expert users.

In our work, we primarily address the needs of communities of d-experts in scientific and technological disciplines. These communities are characterized by different technical methods, languages, goals, tasks, ways of thinking, and documentation styles (Varela, 1979). The members of a community communicate among them through documents, expressed in some notations, which represent (materialize) abstract or concrete concepts, prescriptions, and results of activities. Often, dialects arise in a community, because the notation is applied in different practical situations and environments. For example, technical mechanical drawings are organized according to standard rules which are different in Europe and in USA (ISO 5456). Explicative annotations are written in different national languages. Often the whole document (drawing and text) is organized according to guidelines developed in each single company. The correct and complete understanding of a technical drawing depends on the recognition of the original standard as well as on the understanding of the national (and also company developed) dialects.

Recognizing users as d-experts means recognizing the importance of their notations and dialects as reasoning and communication tools. It also suggests to develop tools customized to a single community. Supporting co-evolution requires in turn that the tools developed for a community can be tailored by its members to the newly emerging requirements (Mørch and Mehandjiev, 2000). Tailoring can be performed only after the system has been released and therefore when it is used in the working context. In fact, a contrast often emerges between the user working activity, which is situated, collaborative and changing, and the formal theories and models that underlie and constitute the software system. This contrast can be overcome by allowing users to adapt themselves the system they are using.

Recognizing the diversity of users calls for the ability to represent a meaning of a concept with different materialization, e.g. text or image or sound, and to associate to a same materialization a different meaning according, for example, to the context of interaction. For instance, a same interface of a distributed system in the automation field, is interpreted in different ways by a technician and a worker. These two d-experts are however collaborating to get a common goal. For this, they use a same set of data, which is however represented according to their specific skills. This is a common case: often experts work in a team to perform a common task. The team might be composed by members of different sub-communities, each sub-community with different expertise. Members of a sub-community should need an appropriate computer environment, suitable to them to manage their own view of the activity to be performed.

When working with a software application, d-experts feel the need to perform various activities that may even lead to the creation or modification of software artifacts, in order to get a better support to their specific tasks, thus being considered activities of EUD. The need of EUD is a consequence of user diversity and user evolution we have discussed. Moreover, the interactive capabilities of new devices have created the potential to overcome the traditional separation between end-users and software developers. New environments able to seamlessly move between using and programming (or customizing) can be designed.

Within EUD, we may include various tailoring activities. Indeed, tailoring activities are defined in different ways in the literature; they include adaptation, customization, end-user modification, extension, personalization, etc. These definitions partly overlap with respect to the phenomena they refer to, while often the same concepts are used to refer to different phenomena. In (Wulf, 1999), tailorability is defined as the possibility of changing aspects of an application's functionality during the use of an application, in a persistent way, by means of tailored artefacts; the changes may be performed by users that are local experts. Tailorability is very much related to adaptability. Different meanings are associated to tailorability and adaptability. To avoid ambiguity, two classes of d-expert activities have been proposed in (Costabile et al., 2003):

Class 1. It includes activities that allow users, by setting some parameters, to choose among alternative behaviors (or presentations or interaction mechanisms) already available in the application; such activities are usually called parameterization or customization or personalization.

Class 2. It includes all activities that imply some programming in any programming paradigm, thus creating or modifying a software artifact. Since we want to be as close as possible to the human, we will usually consider novel programming paradigms, such as programming by demonstration, programming with examples, visual programming, macro generation.

While many systems exist which support the performance of activities of class 1, our EUD methodology aims at the development of systems which allow d-experts to perform activities of class 2, as we will see in the following sections.

4. Software Shaping Workshops

The methodology we have developed to design visual interactive systems considers the following features: 1) adopting the user notation in the system development; 2) offering different views of the activity to the various members of the same community; 3) allowing end-users to participate to system tailoring; 4) guaranteeing a gentle slope of complexity (Myers et al., 2003). The latter means that, in order to be acceptable by its users, the system should avoid big steps in complexity and keep a reasonable trade-off between ease-of-use and expressiveness. Systems might offer for example different levels of complexities, going from simply setting parameters to integrating existing components, up to extending the system by programming new components. To feel comfortable, users should work at any time with a system suitable to their specific needs, knowledge, and task to perform. To keep the system easy to learn and easy to work with, only a limited number of functionalities should be available at a certain time to the users, those that they really need and are able to understand and use. The system should then evolve with the users, thus offering them new functionalities only when needed.

More precisely, our approach to the design of visual interactive systems for specific communities of d-experts is to organize a system as composed of various environments, each one devoted to a specific sub-community. Such environments are organized in analogy with the artisans workshops, where the artisans find all and only the tools necessary to carry out their activities. Following the analogy, d-experts using a virtual workshop find available all and only the tools required to develop their activities. These environments are called *application workshops*, because they allow d-experts to perform their daily tasks.

Using an application workshop, d-experts of a sub-community can work out data from a common knowledge base and produce new knowledge, which can be added to the common knowledge base. All the data available for the community are accessible by each d-expert using the specialist notation of its sub-community.

The application workshops are designed by a design team composed by various experts, who participate to the design using workshops tailored to them. These workshops are called *system workshops* and are characterized by the fact that they are used to generate or update other workshops. Using a system workshop, some experts of the design team defines notations and tools, which are added to the common knowledge base and made available in the generated workshops.

This approach leads to a workshop hierarchy that tries to bridge the communicational gap between software engineers and experts of the application domain, since all cooperate in developing computer systems customized to the needs of the users communities without requiring them to become skilled programmers.

The system workshop at the top of the hierarchy is the one used by the software engineers. Each system workshop is exploited to incrementally translate concepts and tools expressed in computer-oriented languages into tools expressed in notations that resemble the traditional user notations, and therefore understandable and manageable by users. More precisely, at each level of the hierarchy but the bottom level, people use a system workshop and might create a child workshop tailored to a different type of d-expert.

The hierarchy organization depends on the working organization of the user community to which the hierarchy is dedicated: each hierarchy is therefore organized into a number of levels. The top level (software engineering level) and the bottom level (application level) are always present in a hierarchy. The number of intermediate levels is variable according to the different working organization of the user community to which the hierarchy is dedicated (Carrara et al., 2002b) and to guarantee a gentle slope of complexity.

Both application and system workshops are *Software Shaping Workshops* (SSWs): interacting with them, users get the feeling of simply manipulating the objects of interest in a way similar to what they might do in the real world. They are 'shaping' software, in that: a) by using a system workshop, d-experts actually create a software artifact, without writing any textual program code, but using high level visual languages tailored to their needs; b) using an application workshop d-experts adapt the appearance and behavior of the available tools according to their culture, skills, and background.

To make clear the above concepts, in Section 6 we refer to a prototype under study in the system automation field, designed to support different communities of workers and technicians.

5. A view on visual interaction

To develop a Visual Interactive System organized as SSW hierarchy, software engineers and d-experts have first to define the pictorial and semantic aspects of the Interaction Visual Languages through which users interact with workshops. In our approach, we capitalize on the theory of visual sentences developed by the Pictorial Computing Laboratory (PCL) and on the model of WIMP (Windows, Icons, Menus, Pointers) interaction it entails (Bottoni et al., 1999). From this theory, we derive the formal tools to obtain the definition of Interaction Visual Languages.

The PCL model recognizes the interaction between users and interactive systems as a syndetic process in which systems of different nature (the cognitive human - the 'mechanical' machine) cooperate to achieve a task (Barnard et al., 2000). The different systems interact by communicating, interpreting and materializing sequences of messages at successive instants of time. If we restrict to the case of WIMP interaction (Dix et al., 1998), the messages exchanged are the whole images which appear on the screen display of a computer and are formed by text, icons, graphs, pictures, windows. Two interpretations of each element on the screen and of each action arise during the interaction: one performed by the user, depending on his/her role in the task, as well as on his/her culture, experience, and skills, and the second internal to the system, associating the image with a computational meaning, as determined by the programs implemented in the system (Mussio, 2003). The user identifies some subsets of pixels on the screen as functional or perceptual units, called *characteristic structures* (**cs**s). Examples of **cs**s are letters in an alphabet, symbols or icons. Users associate to each **cs** a meaning: the association of a **cs** with a meaning is called *characteristic pattern (***cp***)*. Users recognize complex **cs**s formed by more simple ones (words formed by letters, plant maps formed by icons etc.) and attribute them a meaning stemming from the meaning of the components **cs**s. The interactive system itself is interpreted as a meaningful entity, a complex **cp**.

From the machine point of view, a `cs` is the manifestation of a computational process that is the result of the computer interpretation a program P. (Note that words in **bold** characters denote entities perceived and interpreted by the human user, while those in `courier` characters denote processes and events perceived, computed and materialized by the computer). The computer interpretation of P creates and maintains active an entity, that we call *virtual entity* (`ve`). Actually, P is a set of programs, some of which - called I (Input) programs - acquire the input events generated by the user actions, some - called AP (APplication) program - compute the `ve` reactions to these events, and some - called O (output) - output the results of this computation.

The program AP (APplication program) must be defined by the `ve` designer, who needs to describe the `ve` dynamics. At each instant, the `ve` state is defined as a *characteristic pattern* `cp=<cs, u, <intcs, matcs>>`, where `intcs` (interpretation) is a function, mapping the current `cs` of the `ve` to the computational state `u` of the program AP and `matcs` (materialization) a function mapping `u` to `cs`.

On the human side, the user interacts with the `ve` in state `cp`, by 1- interpreting the **cs** on the screen, 2- manifesting his/her intention by an action **ac**= <operation, **cs**>, operating on the input devices of the machine such as keyboards or mice. The input program I captures the input event generated by the action **ac**, relates this event to a known characteristic structure `cs`, a subset of the `cs` on the screen, and translates it into an input to AP. AP receives these inputs and computes the response to the human activity evolving `u` into a new state `u'`. The results of this computational activity are sent to O, which materialize them as new `cs`s which modify the image maintained visible on the screen by the system. In this way the `ve` reaches a new state `cp'`.

The interaction is *adequate* if a) the **cs** recognized by the human on the screen – i.e. in the current image – matches the `cs` known by the system, and b) the interpretation of the human models in a plausible way the computational meaning `u` - i.e. the reaction of the interactive system is the one expected by the users and understandable by them.

A simple example of `ve` is the “floppy disk” widget to save file in the standard toolbar of MS Word™. This `ve` has different materializations to indicate different states of the computational process resulting from the interpretation of P: for example, once it is clicked by the user the disk shape is highlighted and the associated computational process saves in a disk file the current version of the document. Once the document is saved, the disk shape goes back to its usual materialization (not highlighted).

Virtual entities extend the concept of widgets (as the case of disk widget before) and virtual devices (Preece, 1994), being more independent from the interface style and including interface components possibly defined by users at run time. Interactive systems implemented following our approach permit new forms of tailoring, which distinguish them from traditional ones, such as Visual Basic scripted buttons in MS Word™. For example, users can add at run-time new widgets to the repertoire made available by the system. These widgets have a computational meaning defined according to the context and the task being performed. In (Costabile et al., 2002) the creation of such a ve in a medical domain is discussed: the user (a radiologist) surrounds a set of pixels tracing a closed curve defining a new **cs**, and associates an annotation to the identified area. The system recognizes the surrounded area as a cs, assigns to it a computational meaning, so defining a new ve.

An interactive system is an environment constituted by virtual entities interacting one another and with the user through the I/O devices.

The user sees the system as a whole ve, whose computational state u is materialized at each instant as an image i on the screen. The designer describes this association as a triple vs=<i, u, <int, mat>>, where i is the array of pixels constituting the current image, u is a suitable description of the current state of the process determining the reaction of the whole system to user activities, int and mat are two functions relating elements of i with components of u. This triple is called *visual sentence* (vs), and it specifies the state of the whole virtual entity (i.e. the whole system).

The designer specifies the dynamics of the system by specifying the initial visual sentence vs_0, the one that is instantiated when the user first accesses the system, and a set of transformation rules that specify how a vs evolves into a different one in reaction to user activities (Carrara et al., 2002b, Fogli et al., 2002).

6. Building SSWs in a real case

In this Section, we provide an example of applying the SSW methodology to a real case we have developed with ETA Consulting, a company producing systems for factory automation. ETA is also responsible of producing the operating software (and related user interface) for the systems that it sells.

6.1 Analysis

ETA Consulting has the following needs: 1) creating systems for factory automation that are usable, i.e. easy to learn and easy to use for its clients; 2) having software tools which support ETA personnel (d-experts) in the development, testing, and maintenance of such systems. As we will describe in the following, ETA personnel is composed of different categories of people with different skills, who need to perform various tasks with the software tools. In accordance with our approach, specific software environments (SSWs) must be developed for each category of users. Similarly, ETA clients need different environments specific for their tasks when operating the automation systems. The analysis we have performed with ETA d-experts and clients of ETA automation systems has lead us to foresee a SSW hierarchy structured in four levels (Figure 1):

1) *A system workshop for software engineers*. This is a basic workshop always at the top of the hierarchy since it is the one used by the team of software engineers, in which they find all tools, programming languages, etc. they need for generating the SSWs for specific applications. Using this workshop, the team defines the libraries of methods for `cs`s creation, the window system (Myers, 1995), the templates for linking `cs`s and elements of the window system, and the Interaction Visual Language, which allows also the ETA technicians (d-experts) to manage all this stuff at level 2.

2) *A system workshop for virtual entity generation*. The software engineers have created this workshop to be used together with ETA d-experts in a kind of participatory design, for generating all `ve`s necessary to the ETA d-experts to develop the systems they sell to their clients. A deep analysis of user requirements has been performed. More specifically, we have analyzed the company and the people working in the company, the kind of applications they develop, their usual clients, the notations and tools they traditionally use to develop their applications, in order to identify the interaction visual languages for this SSW. The `ve`s created in this workshop represent the tools necessary to ETA d-experts for their activities. We identified two main activities of ETA d-experts: the first one related to the software mechanical design and testing of the automation system; the second one referring to the automation system operating in the client factory (see Figure 1). Consequently, once all `ve`s are created, two child workshops are generated: the first used by ETA d-experts for creating

environments suitable for the first activity; the other for creating the applications for the clients.

3) *One or more system workshops for application workshop creation.* Given the `ve`s made available by the system workshop at level 2, the ETA d-experts (technicians) use the workshop at this level to generate the application workshops for the other d-experts or for the end-users. They compose various prototypes of the application workshop by selecting the `ve`s prepared at level 2. In accordance with a user-centred approach, such prototypes are evaluated together with the other d-experts and end-users in order to choose the most appropriate for them.

4) *One or more application workshops* devoted to the different professionals working at ETA, as testers (d-experts), or in the client factory, as operators of the developed application. More in detail, in ETA there are mechanical designers and testers, software designers and testers. Therefore, we identified for them three different application workshops: the first for mechanical testing, the second for software testing, and the last for mechanical programming of the automation systems. Besides, among ETA clients, who use the automation systems produced by ETA, we found other two kinds of users: assembly-line operators and production managers. In this case other three application workshops have been identified: one for operators and the other two for managers.

The intermediate levels in the hierarchy are developed to cope with the need to gradually adapt the systems to the complexity of the tasks (gentle slope of complexity).

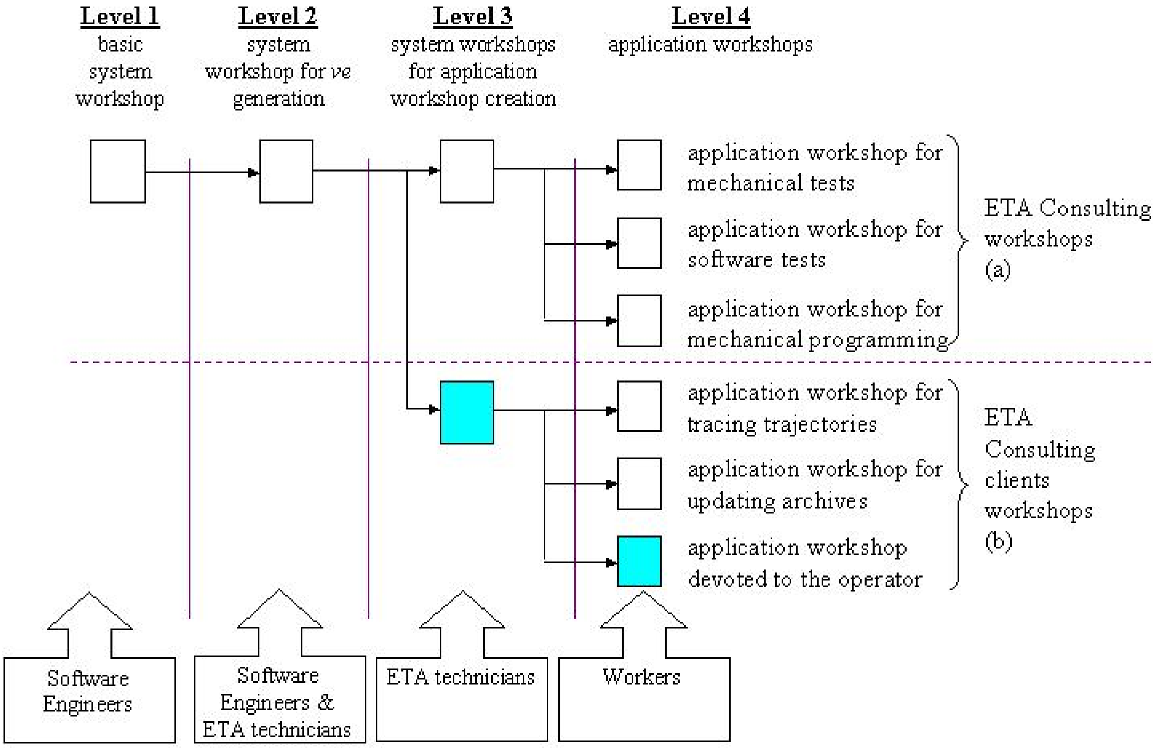


Fig. 1: The workshops hierarchy in the case of ETA Consulting.

6.2 The workshops developed for the ETA case study

At the moment, we have developed two prototype workshops for the ETA case study (Fogli et al., 2003). They are the application workshop devoted to the operator and the system workshop permitting the mechanical engineers to create the application workshop. In Figure 1, the rectangles corresponding to such workshops are highlighted.

The application workshop is devoted to the control of a pick-and-place robot. The required functionalities of this system were: different modalities of using the robot (automatic, manual, diagnostic, setting, etc.), the possibility to choose among various tools to be associated to the robot to modify its behavior and the task to perform, and finally a number of options to put annotations, to require an automatic help, to save the work, etc.

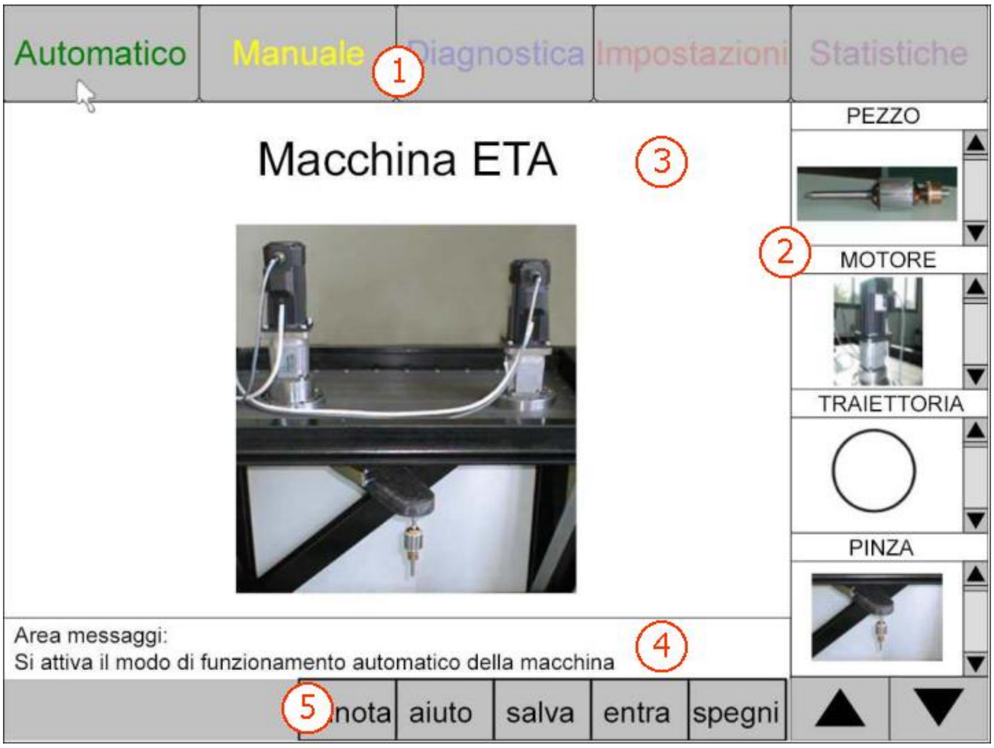


Fig. 2: The application workshop developed for ETA Consulting (the numbers have been added on the screenshot for the sake of explanation in the text).

A prototype of such software environment has been then developed, and its initial state is shown in Figure 2. The robot operation modality may be chosen by clicking on one of the button in the button panel indicated by the number 1 in the figure; the tools to be associated to the robot may be selected from the archives of pieces, engines, trajectories and grippers shown on the right part of the interface (2); the behavior of the machine is then shown in the working area indicated by the number 3 in the figure; finally, at the bottom, a message area (4) presents messages orienting the user during his/her interaction with the system and the button panel (5) offers the options of annotation, help, saving, logging, exit.

In the case at hand, an example of `ve` is the button “Automatico”: this `ve` has different materializations to indicate different states (characteristic patterns) of the computational process generating the `ve`: for example, once it is clicked by the user both the text and the background changes color to give a feedback to the user; moreover, the associated computational process runs the machine in the automatic modality. If the user successively clicks on another button to request a different operation modality, the colors of button “Automatico” go back to their default values (red text and gray background), the other selected button changes its colors, and the machine stops running in the automatic modality.

To create the application workshop described above, we have developed a prototypal system workshop for mechanical engineers which supports them in the creation of the application workshop through simple drag-and-drop activities. Figures 3 and 4 illustrate two different snapshots of the system workshop. The snapshots are generated during the interaction of the domain-expert with the system workshop for creating the application workshop shown in Figure 2. In Figure 3, the virtual entity “canvas” has been already selected from the menu area on the right side (1) and positioned on the working area (2) to become the background of the system that is being created. The mechanical engineer is now dragging and dropping the virtual entity “bottoniera operativa” (operative button panel). This button panel has also been selected from the menu area and can be positioned on the top of the canvas to become the area where the buttons “Automatico”, “Manuale”, etc. can be successively positioned (see Figure 2). In Figure 3, it can also be seen a message area (3), suggesting the user the operation to do in his own language, and a button panel (4) containing the tools that support the mechanical engineer in the saving or loading process, in the simulation of the system being created, in annotating the environment, and so on. Finally, a virtual entity playing the role of title (5) is also present at the top of the environment, being a cornerstone for the domain-expert.

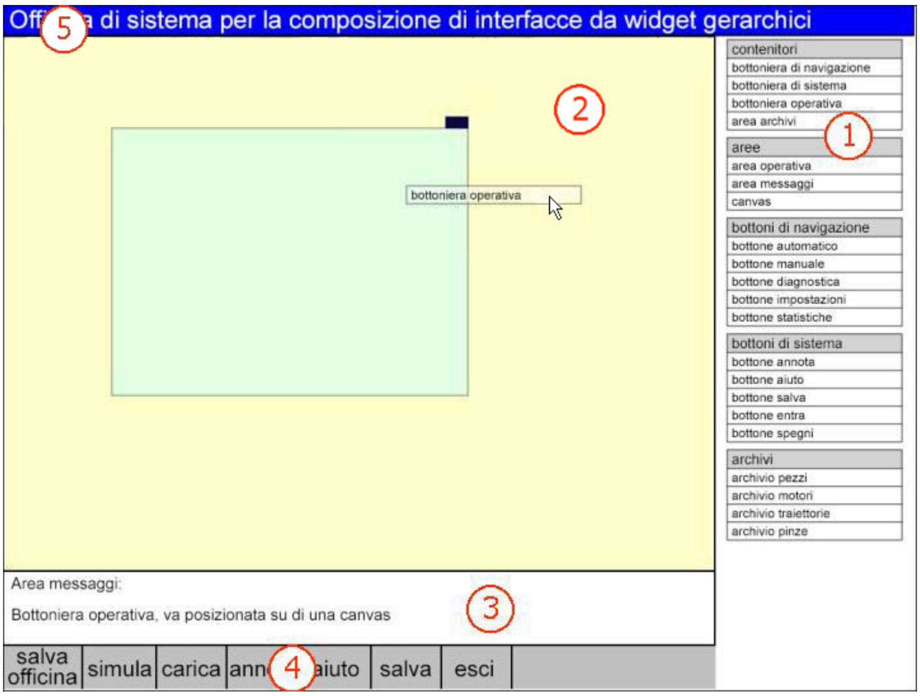


Fig. 3: The virtual entity “bottoniera operativa” (operative button panel) is dragged and dropped on the canvas representing the background of the workshop being created (the numbers have been added on the screenshot for the sake of explanation in the text).

Figure 4 shows the application workshop presented in Figure 2 partially composed. The d-expert is now positioning a new button on the operative button panel. Note that the virtual entity corresponding to the operative button panel is present in both snapshots shown in figures 3 and 4. However, the virtual entity is in two different states, represented by two different `cp`s: the `cp` in Figure 3 has an associated characteristic structure corresponding to the white bar with a text inside; while, in Figure 4, the characteristic structure is the gray bar over which three buttons have already been positioned.

Fig. 4: A part of the application workshop has been created. The button “diagnostica” (diagnostic) is being located on the operative button panel.

6.3 Implementation

The implementation is based on the techniques and tools made available within the W3C framework. A workshop is implemented as an IM^{2}L program. IM^{2}L (Interaction Multimodal Markup Language) is an XML-based language that provides the rules for the definition of virtual entities: its markup tags encode a description of the possible `ve`s to be used in the application at hand (Salvi, 2003). An IM^{2}L program is composed by a set of XML-based documents and a library of javascript functions. It runs under a common web browser, enriched by the Adobe SVG Viewer plugin. SVG is the XML specification for vector graphics (W3C, 2001) and is exploited to specify the `ve`s materialization.

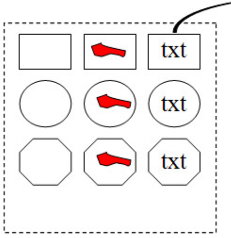
An IM^{2}L program implementing a system workshop can be steered by its users to self-transform into a new IM^{2}L program, which can results into a further system workshop or into an application workshop. This self-transformation property is used to generate a SSW hierarchy. On the whole, a SSW hierarchy is generated from the system workshop of the software engineer by an incremental process determined by the activities of the experts of the design team. More details can be found in (Fogli et al., 2003).

In order to illustrate the creation and specialization of `ve`s necessary to the ETA environments, we describe the definition and creation of the `ve` "button" at the different levels of the hierarchy, by adopting the above mentioned implementation techniques.

Level 1. Using the SSW at level 1, the software engineer provides the IM^{2}L definition of the type "button", as shown in Figure 5. This is an XML-based description of the logical structure of a button. Most of the attributes (e.g. 'id', 'shape', 'oncl', 'onover', etc.) contain a value that will be instantiated at the next steps of the button creation (levels 2 and 3). Then, the software engineer defines a library CS_j of possible button shapes as a set of SVG prototypes. A javascript function must also be defined by the software engineer to transform the IM^{2}L description of the button into an instance of the SVG prototype. Figure 6 shows an example of a library of `cs`s and the SVG prototype for a button having a rectangular shape and a textual label (the characteristic structure located at the top-right of the square). Moreover, the software engineer creates a library U_j of javascript functions defining the computations to be associated to a button, including standard computations typical of a WIMP system (for example open a window when clicking on button), and application-oriented computations, i.e. related to the automation system operation in the ETA case study. Finally, the IM^{2}L definition, the SVG prototypes and the javascript functions are made available to the next level in the hierarchy (level 2).

```
<button
        template="yes"
        id="buttonIdentifier"
        position="0,0"
        dimension="0x0"
        shape="buttonShape"
        color="buttonColor"
        stroke-width="1"
        oncl="functionOnClick"
        onover="functionOnMouseOver"
        onout="functionOnMouseOut"
        <text position="0,0"
                fill="black"
                font-size="20">
                ButtonText
        </text>
        <text id="buttonIdentifierD">
                Description button
        </text>
</button>
```

Fig. 5: An example of IM^{2}L definition for a `ve` of type “button”.

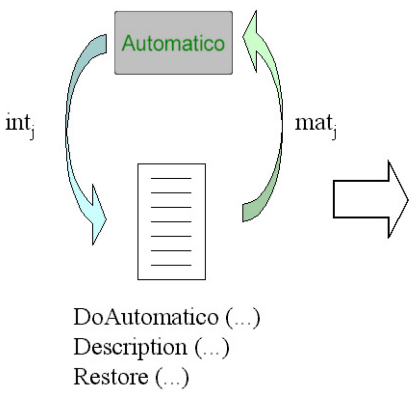



```
<g id="button"
        transform="translate(0 0)"
        flag="0"
        ondown=""
        onup=""
        onmove= ""
        onmousedown=""
        onmouseup=""
        onmousemove=""
        clone="0" drag="1" select="1">
        <rect id="" width="" height=""
                rx="0" ry="0"
                fill="rgb(0,0,0)"
        stroke="rgb(0,0,0)" stroke-
        width="1"/>
        <text transform="translate(0 0)"
        fill="rgb(0,0,0)" font-size="0"
        font-family="Arial"> </text>
        <desc id=""> </desc>
</g>
```

Fig. 6: The library $\mathtt{cs}_j$ of button shapes and an example of SVG prototype for the rectangular one.

Level 2. At this level, the ETA d-expert associates a button shape (a characteristic structure) cs_j with a computation u_j, by defining the pair $<int_j, mat_j>$ obtaining the characteristic pattern $cp_j=<cs_j, u_j, <int_j, mat_j>>$ that specifies the initial state of the `ve` “button”. This association, i.e. the definition of the pair $<int_j, mat_j>$, is done by specifying the attributes in the IM^{2}L description. Some parameters are set at this level while other remain variable, to be set at the next level. As shown in Figure 7, the d-expert sets the following parameters: the button identifier, the shape of the button, the names of the computations associated with the activities performed with the mouse, and a link to a textual description of the button functionalities. All the other attributes assume default values which can be modified at level 3. Note that in the left part of Figure 7, a schematize picture of the characteristic pattern is provided, including the characteristic structure of the button "Automatico" in its initial state, the associated computations "DoAutomatico(...)", "Description(...)", "Restore(...)" (for the sake of simplicity, we do not show here the complete signatures of the functions), and the links between the characteristic structure and the computations, i.e. int_j and mat_j.

The created characteristic patterns specifying buttons are then organized in a button library to be made available to the workshop at level 3.

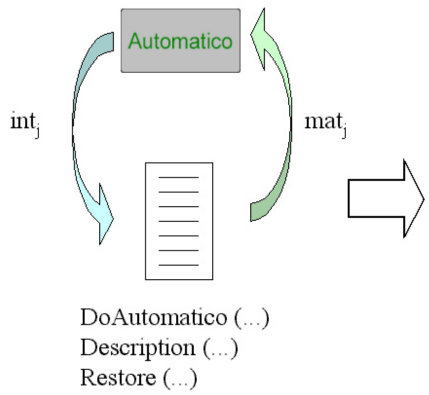



```
<button
      template="yes"
      id="button_automatico"
      position="0,0"
      dimension="160x80" shape="rect"
      color="black"
      stroke-width="1"
      oncl="DoAutomatico, evt"
      onover="Description, 'automaticoD'"
      onout="Restore"
      <text position=
            "translate(10 50)"
            fill="black"
            font-size="20">
            Automatico </text>
      <text id="button_automaticoD">
      </text>
</button>
```

Fig. 7: cp_j definition at level 2: values highlighted in bold are definitively assigned to the attributes.

Level 3. At this level, the d-expert, while composing a specific interface, instantiates the characteristic patterns made available by level 2. The interface composition is visually performed: for example, the values of the attributes “position” and “dimension” are set automatically as a consequence of the positioning activity and the run-time adjustment of the button dimension (see also the example previously discussed with Figures 3 and 4). Other parameters, e.g. button color, can be set by the d-expert through a parameter setting facility accessible by clicking on the button. Figure 8 shows the final definition of the button: the values, which are specified at this level of the hierarchy, are in bold.

Level 4. At this level, the end-user uses the application workshop generated by the system workshop at level 3 to carry out his/her task. In the example case (see Figure 2), s/he may interact with the button “Automatico” to start the machine in the automatic modality. In summary, the `ve` button “Automatico” is incrementally defined in shape, content, and behavior throughout levels 1-3 to be used at level 4.




```
<button
      template="yes"
      id="button_automatico"
      position="0,0"
      dimension="160x80" shape="rect"
      color="lightgrey"
      stroke-width="1"
      oncl="DoAutomatico, evt"
      onover="Description, 'automaticoD'"
      onout="Restore"
      <text position=
            "translate(10 50)"
            fill="green"
            font-size="28">
            Automatico </text>
      <text id="button_automaticoD">
      Machine automatic modality
      is activated </text>
</button>
```

Fig. 8: cp_j instantiation at level 3: the values in bold are definitively specified.

7. Conclusions

Most users require environments in which they can make some ad hoc programming activity related to their tasks and adapt the environments to their emerging new needs. Moreover, user-system interaction is currently difficult for several reasons, including the user diversity and the co-evolution of systems and users. The methodology discussed in this paper is a step toward the development of powerful and flexible environments, with the objective of easing the way users interact with computer systems to perform their daily work. The methodology can be exploited to generate User Interface Development Environments (UIDEs) to be used by different user communities. Such environments are specialized to the community culture, background, skills and needs. It is a participatory approach, in that people belonging to a particular user community may participate to the development of the software environments devoted to the community itself. More precisely, a system workshop is a UIDE specialized to a given user community, which allows a designer to develop, generate and test SSWs devoted to other users. Whenever the developed SSW is devoted to an end-user, it is an application workshop, and it allows the user to perform an application task. If the workshop is devoted to a designer (a software engineer, an HCI expert or a domain expert), it is in turn a system workshop.

All the SSWs have the same structure. When a system workshop is used to create a new workshop, a portion of the system workshop is replicated as the kernel of the new workshop (Fogli et al., 2003).

8. Acknowledgements

We are grateful to Denise Salvi who developed the prototype, and to Silvano Biazzi of ETA Consulting (silvano.biazzi@cjb.it) for providing the case study.

The support of EUD-Net Thematic Network (IST-2001-37470) is acknowledged.

9. References

Arondi, S., Baroni, P., Fogli, D., Mussio, P. (2002). Supporting co-evolution of users and systems by the recognition of Interaction Patterns, Proc. of the International Conference on Advanced Visual Interfaces (AVI 2002), Trento, Italy, pp. 177-189.

Barnard, P., May, J., Duke, D., Duce, D. (2000). Systems, Interactions, and Macrotheory. ACM Trans. on Human-Computer Interaction, 7(2), pp. 222-262.

Boehm, B. W., Abts, C., Brown, A.W., Chulani, S., Clark, B.K., Horowitz, E., Madachy, R., Reifer, D.J. and Steece, B. (2000). Software Cost Estimation with COCOMO II. Prentice Hall, Upper Saddle River, NJ.

Bottoni P., Costabile M.F, Mussio P. (1999). Specification and Dialog Control of Visual Interaction. ACM Transactions on Programming Languages and Systems, 21(6), 1077-1136.

Bourguin, G., Derycke, A., Tarby, J.C. (2001). Beyond the Interface: Co-evolution inside Interactive Systems. Proc. IHM-HCI 2001, Lille, France.

Carrara, P., Fogli, D., Fresta, G., Mussio, P. (2002a). Making Abstract Specifications Concrete to End-Users: the Visual Workshop Hierarchy Strategy. Proc. 2002 IEEE Symposia on Human Centric Computing (HCC'02), Arlington (VA), USA, pp. 43-45.

Carrara, P., Fogli, D., Fresta, G., Mussio, P. (2002b). Toward overcoming culture, skill and situation hurdles in human-computer interaction. Int. Journal Universal Access in the Information Society, 1(4), pp. 288-304.

Costabile, M.F., Fogli, D., Fresta, G., Mussio, P., Piccinno, A. (2002). Computer Environments for Improving End-User Accessibility. Proc. of 7th ERCIM Workshop "User Interfaces For All", Paris, France, pp. 187-198.

Costabile, M.F., Fogli, D., Letondal, C., Mussio, P., Piccinno, A. (2003). Domain-Expert Users and their Needs of Software Development. Proc. Special Session on EUD, UAHCI Conference, Crete, Greece, pp. 532-536.

Cypher, A. (1993). Watch What I Do: Programming by Demonstration, The MIT Press, Cambridge.

Dix, A., Finlay, J., Abowd, G., Beale, R. (1998). Human Computer Interaction, Prentice Hall, London.

Fogli, D., Mussio, P., Celentano, A., Pittarello, F. (2002). Toward a Model-Based Approach to the Specification of Virtual Reality Environments. Proc. IEEE International Symposium on Multimedia Software Engineering (MSE'2002), Newport Beach (CA), USA, pp. 148-155.

Fogli, D., Piccinno, A., Salvi, D. (2003). What Users See Is What Users Need, Proc. Distributed Multimedia Systems (DMS'03), Miami (FL), USA, 335-340.

ISO 5456: ISO Standard Technical Drawing Projection Methods.

Mørch, A. I., Mehandjiev, N. D. (2000). Tailoring as Collaboration: The Mediating Role of Multiple Representations and Application Units, Computer Supported Cooperative Work, 9, pp. 75-100.

Myers, B. A. (1995). User Interface Software Tools, ACM Transactions on Computer-Human Interaction, Vol. 2, No. 1, pp. 64-103.

Myers, B., Hudson, S. E., Randy, P. (2003). Past, Present, and Future of User Interface Software Tools, Human-Computer Interaction in the New Millennium, Carroll (ed.), Addison-Wesley.

Mussio, P. (2003). E-Documents as tools for the humanized management of community knowledge. Keynote Address, Proc. ISD 2003, Melbourne, AUS.

Nielsen, J. (1994). Usability Engineering, Academic Press, San Diego.

Preece, J. (1994). Human-Computer Interaction, Addison-Wesley.

Salvi, D. (2003). Progettazione di ambienti integrati per la produzione di ambienti interattivi, Laurea Thesis, Università di Brescia, Italy.

Varela, F. J. (1979). Principles of Biological Autonomy, GSR Amsterdam, North Holland, 1979.

W3C: Scalable Vector Graphics (SVG), [Online] 2001 <http: //www.w3c.org/Graphics/SVG/>.

Wulf, V. (1999). Let's see your Search-Tool! - Collaborative use of Tailored Artifacts in Groupware. Proc. of GROUP '99, Phoenix, USA, pp. 50-60.

PsychNology Journal, 2004
Volume 2, Number 1, 123 - 139

Methods and Tools for Designing and Developing Usable Multi-Platform Interactive Applications

Cristina Chesta♠[1], Fabio Paternò[2], Carmen Santoro[2]

(1) Motorola Electronics S.p.A. – Global Software Group
Turin, Italy

(2) I.S.T.I. - C.N.R.
Pisa Italy

ABSTRACT

The increasing availability of new types of interaction devices raises the need for new methods and tools to support the design and development of highly usable context-sensitive nomadic applications accessible through multiple platforms.
This paper provides an overview and discusses a solution based on the use of multiple levels of abstractions, which has been studied within the framework of the European project CAMELEON. Moreover it addresses the problem of evaluating the usability of these tools by discussing the specific issues, the criteria and methodologies applied as well as some results obtained in an experimental activity on the subject.

Keywords: *nomadic, multi-platform, context-aware, model-based*.

Received 23 January 2004; received in revised form 1 April 2004; accepted 6 April 2004.

1. Introduction

With the advent of the wireless Internet and the rapidly expanding market of smart devices, designing interactive applications supporting multiple platforms has become a difficult issue. In fact, on the one hand the decreasing cost at which the devices are now offered has enabled an increasing variety of people to become potential users of features and services of novel generations of communication technology as never before.

On the other hand, rarely such a high number of flourishing range of opportunities offered have become effective, due to the low quality of the user interfaces provided to the users.

♠ Corresponding Author:
Cristina Chesta
E-mail: Cristina.Chesta@motorola.com

The main problem is that many assumptions that have been held up to now about classical stationary desktop systems are being challenged when moving towards nomadic applications, which are applications that can be accessed through multiple devices from different locations. Each device is characterized by different interaction resources, including computational capability, accessible display area, interaction channels and network bandwidth. Moreover the interaction resources are subject to variations according to the physical and environmental conditions. For software developers, this introduces the difficult task of constructing multiple versions of single applications and endowing these versions with the ability to dynamically respond to changes in context. Currently, developers often create different versions of applications for different devices. This requires extra development, and maintenance costs and complicates the configuration management. A proliferation of versions reduces the resources available for usability engineering, and requires expensive maintenance of cross-platform consistency of the user interface.

Consequently, one fundamental issue is how to support software designers and developers in building such applications: in particular, there is a need for novel methods and tools able to support development of interactive software systems able to adapt to different targets while preserving usability.

The evaluation of the tools for nomadic applications development requires specific criteria and methodologies enabling the assessment of usability and effectiveness from the double point of view, of the developer using the tool itself and of the final user dealing with the application implemented by exploiting the tool.

The paper presents some innovative techniques to provide software engineering support for the development of applications accessible through multiple heterogeneous platforms, which have been studied within the framework of the European project CAMELEON.

The paper is organized as follows. We present first a discussion on related work, a comprehensive vision of the project objectives and the approach adopted.

The next sections are dedicated to the current activities, with special focus to the ones carried-out at ISTI-CNR and Motorola GSG Italy. We introduce the TERESA tool for forward engineering design and development of multi-platform applications. Then we illustrate the case study proposed in order to provide a real application example and the experimental evaluation performed. Finally we report some preliminary evaluation results, followed by the concluding remarks.

2. The Approach

In a recent paper discussing the future of user interface tools Myers, Hudson, and Pausch (Myers et al., 2000) indicate that the wide platform variability encourages a return to the study of some techniques for device-independent user interface specification, so that developers can describe the input and output needs of their applications, so that vendors can describe the input and output capabilities of their devices, and so that users can specify their preferences. Then, the system might choose appropriate interaction techniques taking all of these into account. This is also called user interface plasticity (Thevenin et al., 1999). Methods for modelling work context (Beyer & Holtzblatt, 1998) can provide useful information for this type of approach.

The basic idea of how to cope with the current situation of heterogeneity of currently available devices and the need for usable User Interfaces (UIs) is that, instead of having separate applications for each device, which exchange only basic data, there is some abstract description and an environment able to suggest a design suitable for a specific device.

This is the main goal of model-based design and development of interactive applications, which have been considered though not extensively adopted during last decade. Nomadic applications raise new challenges that can be better addressed using a model-based approach. There is a need for a unitary view of nomadic applications, even if their parts require different instantiation for different platforms. This allows designers to understand and control the dependencies among such instances. Secondly, new design criteria suitable for mobile devices should be introduced. The potentialities of these approaches have only begun to be addressed. In the GUITARE Esprit project (http://giove.cnuce.cnr.it/guitare.html) a user interface generator was developed: it takes ConcurTaskTrees (CTT) task models (Paternò, 1999) and produces user interfaces for ERP applications according to company guidelines. However, automatic generation is not a general solution because of many, varying factors that have to be taken into account within the design process. Semi-automatic support is more general and flexible: Mobi-D (Puerta, 1997) is an example of a semi-automatic approach, but it only supports design of traditional graphical desktop applications.

UIML (Abrams, 1999) is an appliance-independent XML user interface language. While this language is ostensibly independent of the specific device and medium used for the presentation, it does not take into account the research work carried out over the last decade on model-based approaches for user interfaces: for example, the language provides no notion of task, it mainly aims to define an abstract structure. The W3C consortium has recently delivered the first version of a new standard (XForms) that presents a description of the architecture, concepts, processing model, and terminology underlying the next generation of Web forms, based on the separation between the purpose and the presentation of a form. If it shows the importance of separating conceptual design from concrete presentation, it also highlights the need for meaningful models to support such approaches.

XIML (Puerta & Eisenstein, 2002) (eXtensible Interface Markup Language, http://www.ximl.org) is an XML-based language, whose initial development took place at the research laboratories of RedWhale Software. It is intended to be a universal user interface specification language, since it provides a way to completely describe a user interface and represent attributes and relations of the important elements of a user interface without worrying about how they will be implemented. In other words, it enables a framework for the definition and interrelation of interaction data items, thereby providing a standard mechanism for applications and tools to interchange interaction data and interoperate within integrated user-interface engineering processes, from design, to operation, to evaluation. Today XIML is probably the most advanced UI specification language, as it can serve for context sensitivity and many other objectives. However, it is worth noting that XIML mainly focuses on syntactic, rather than semantic aspects. In addition, tool support is not publicly available. Collagen (Rich & Sidner, 1998) uses an explicit embedded task model to support the creation of task-aware collaborative agents. The agent interprets and guesses the user's current intentions, and can determine efficient plans to achieve them. The issue related to platforms is not considered.

More generally, the issue of applying model-based techniques to the development of UIs for mobile computers has been addressed at a conceptual and research level (Calvary et al., 2001), (Einsenstein et al., 2001) but there are still many issues that need to be solved to identify systematic, general solutions that can be supported by automatic tools.

The CAMELEON approach aims to support design and development of nomadic applications providing general solutions that can be tailored to specific cases, whereas current practice is still to develop ad hoc solutions with few concepts that can be reused in different contexts.

The actors in charge of adaptation depend on the phase of the development process:

- At the design stage, multi-targeting can be performed explicitly by humans such as system designers and implementers, and/or it can rely on dedicated tools.
- At run time, the adaptation may be performed by the user and/or the system. A UI is adaptable when it adapts at the user's request (typically, by providing preferences menus). It is adaptive when the user interface adapts on its own initiative.

A distinction can also be made between methods for forward engineering, allowing automatic generation of the interface for various targets starting from a common abstract description of the scenario to address, and methods for reverse engineering, which automatically transform web pages to pages at a certain level of abstraction, and these result pages can later on be transferred to other computing platforms.

Currently the forward engineering approach seems to be more promising, even if combining the two approaches by automatically reconstructing a task model from a web page and then automatically converting it for others platforms would open great opportunities from the application point of view.

A set of methods and tools supporting a number of transformations useful when designing multi-platform applications have been proposed by the CAMELEON consortium (Berti et al., 2003). At ISTI the TERESA tool has been developed, currently supporting transformations from task models to desktop, phone user interfaces and vocal interfaces (XHTML, XHTML Mobile Profile, and VoiceXML). Another tool called Web Revenge and supporting automatic reconstruction of task models from HTML code has been developed as well. A different approach to reverse engineering of web sites has been investigated at University of Louvain where the VAQUITA and RUTABAGA tools have been implemented supporting reconstruction of presentation models from HTML code. Unlike the previous tools, ArtStudio, developed at University of Grenoble, allows development of Java interfaces for multi-platform applications. Research activities are also ongoing about run-time mechanisms and infrastructure (Coutaz et al., 2003).

The set of tools demonstrate how the concepts and methods developed can be incorporated in real tools that can support the work of designers and developers in many types of software companies.

As far as the integration between the CAMELEON tools is concerned, effort has been put within the consortium particularly on the communication between two tools: VAQUITA and TERESA. Since the first one mainly covers an abstraction step while the second one covers reification, an example of interest for the consortium was judged to analyse the result of a two-step process in which e.g. the output of reverse-engineering a web page in VAQUITA (first step: abstraction) becomes the input for TERESA tool to the aim of performing in turn a reification step on it and possibly re-design the user interface for another computing platform.

Such an integration has been achieved through the introduction of a common XML-based language, CameleonXML (Limbourg et al., 2004), used to describe abstract user interface and developed by the consortium having in mind the general goal of modelling and represent the different requirements about the design of multi-platform user interfaces that have been raised up to now by discussions within the project.

In order to validate the CAMELEON approach and to elicit requirements for the tools design, the industrial partners provided examples of application to real case studies. In particular Motorola GSG Italy proposed an *e-Desk* service allowing people to access from any place with different devices office productivity applications, including an *e-Agenda* offering calendar, appointment schedule and automatic reminder (Chesta & Fliri, 2003).

The multi-context interface of both e-Desk service and e-Agenda application has been realized through the support of TERESA tool, serving as basis for the experimental evaluation (Chesta et al, 2003).

3. The TERESA Tool

TERESA is a transformation-based environment supporting a number of transformations useful for designers to build and analyse their design at different abstraction levels and consequently generate the concrete user interface for a specific type of platform (Paternò, 1999), (Paternò & Santoro, 2003), (Mori et al., 2003).

The abstraction levels considered (see Figure 1 at the end of this section) are:

- *High level task modelling*: the output of this phase consists of the description of the logical activities that need to be performed in order to reach the users' goals. This description initially considers an integrated task model where all the activities that have to be supported have been specified. Next, the task model is refined and structured so as to identify the activities that have been supported for each platform considered. For example, a nomadic task model could analyse the activities supported by a system for reserving a hotel room through a cell phone and a desktop system: such specifications might share portions of the task model (the activities are performed in the same way), while being different for other tasks (for example some details about the room might be neglected with the cell phone platform);

- *Abstract user interface (AUI):* in this phase the focus shifts to the interaction objects supporting task performance. After having obtained the task model for a specific platform, an abstract user interface is derived from it. It is defined in terms of presentations (the set of user interface elements perceivable at the same time), and each presentation is composed of a number of interactors (Paternò & Leonardi, 1994), which are abstract interaction objects identified in terms of their main semantics effects. For instance, going on with the hotel reservation example, at this level we will just consider that for selecting a specific hotel, we do need some widget supporting a *single selection* task: the implementation details of such an object are irrelevant at this moment, and for this reason we identify it as an *abstract* interaction object supporting a selection. An XML-based language has been specified in order to describe the organisation of the various interactors within the presentations. The structure of the presentation is defined in terms of elementary interactors characterised in terms of the task they support, and their composition operators. Such operators are classified according to the communication goals to achieve: a) Grouping: indicates a set of interface elements logically connected to each other; b) Relation: highlights a one-to-many relation among some elements, one element has some effects on a set of elements; c) Ordering: some kind of ordering among a set of elements can be highlighted; d) Hierarchy: different levels of importance can be defined among a set of elements.

- *Concrete user interface (CUI*): at this point each abstract interactor is replaced with a concrete interaction object depending on the type of platform and media available and with a number of attributes that define more concretely its appearance and behaviour. For example, the abstract interaction object we mentioned in the previous phase (an interactor supporting the selection of a specific hotel) could be rendered through a scrollable list-box on a desktop platform and through a pull-down menu on a cell phone platform. It is worth pointing out that, at this level, there is still no mention about the specific language used for implementing such concrete objects.

- *User interface generation*: this phase is completely platform-dependent and has to consider the specific properties of the target device. The interactors are mapped into interaction techniques supported by the particular device configuration considered (operating system, toolkit, etc.), and also the operators defined in the language for abstract user interface are implemented with appropriate presentation techniques. At this level we should specify e.g. if the pull-down menu on the cell phone platform will be rendered through WML, or through XHTML Mobile Profile, etc..

A number of main requirements have driven the design and development of TERESA:

- *Mixed initiative*: we want a tool able to support different level of automation ranging from completely automatic solutions to highly interactive solutions where designers can tailor or even radically change the solutions proposed by the tool.

- *Model-based*: the variety of platforms increasingly available can be better handled through some abstractions that allow designers to have a logical view of the activities to support, then the call for effective models able to capture the relevant information that should be considered.

- *XML-based* languages have been proposed for every type of domain. In the field of interactive systems there have been a few proposals that partially capture the key aspects to be addressed.

- *Top-down*: this approach is an example of forward engineering. Various abstraction levels are considered, and we support cases when designers have to start from scratch. So, they first have to create more logical descriptions and then move on to more concrete representations until the final system.

- *Different entry-points*: our approach aims to be comprehensive and to support the entire task/platform taxonomy. However, there can be cases where only a part of it needs to be supported.

- *Web-oriented*: the Web is everywhere; therefore, for generality purposes, we decided that Web applications should be our first target. However, the approach can be easily extended to other environments (such as Java applications, Microsoft environments, etc.) because only the last transformation needs to be modified for this purpose.

The TERESA tool offers a number of transformations between different levels of abstractions and provides designers with an easy-to-use integrated environment for generating both XHTML and VoiceXML user interfaces (Berti & Paternò, 2003). With the TERESA tool, at each abstraction level the designer is in the position of modifying the representations while the tool keeps maintaining forward and backward the relationships with the other levels thanks to a number of automatic features that have been implemented (e.g. the possibility of links between abstract interaction objects and the corresponding tasks in the task model so that designers can immediately identify their relations). This result is a great advantage for designers in maintaining a unique overall picture of the system, with an increased consistence among the user interfaces generated for the different devices and consequent improved usability for end-users.

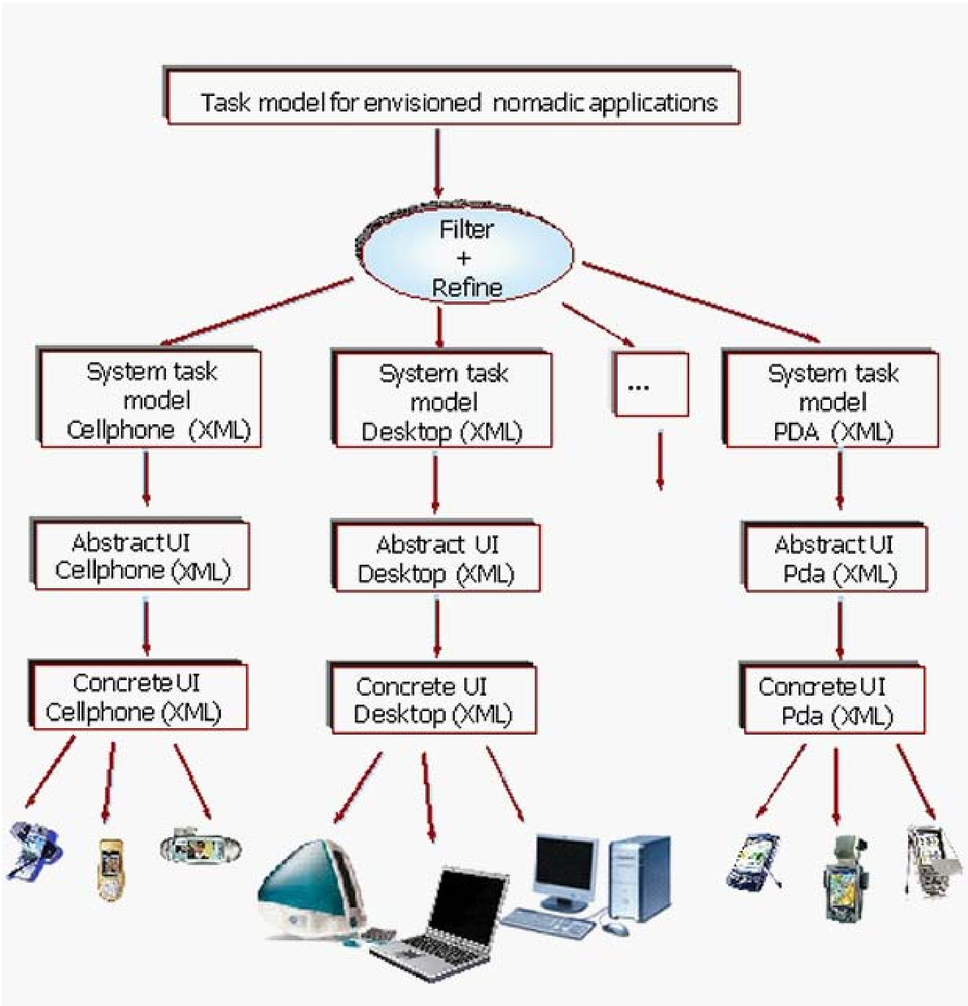


Fig.1: The Main Abstraction Levels of TERESA

4. Experimental Evaluation

The experimental evaluation has been conducted in parallel to the tool development in order to provide a formative rather than a summative evaluation.

Starting with the ISO 9241-11 standard definition (ISO9241-11, 1991) and Shneiderman's (Schneidermann, 1998) and Nielsen's (Nielsen, 1994) metrics, but considering the double perspective of the tool itself versus the product realized through the tool, we identified four aspects to be evaluated and eight related requirements as listed in Table 1.

Two experiments have been designed in order to cover different aspects according to the criteria framework formerly exposed. Both of them refer to the common application scenario related to Business to Employee environment.

Aspect	***Requirement***
Tool Interface	Intuitiveness
	Learnability
Tool Functionalities	Completeness
	Developer satisfaction
Final Product Obtained by employing the Tool	User Satisfaction
	Maintainability and Portability
Approach Cost/Effectiveness	Development Efficiency
	Integrability

Table 1: Evaluation criteria

Five subjects, selected within Motorola GSG Italy staff, were involved in the evaluation. All of them, within a range of different background and specialization, have technical knowledge and experience in software design and development, and are experienced computer users. They have been asked to participate in a 30 minutes preparation session and to dedicate 10 minutes reading the TERESA help prior to start the exercises.

The first experiment focused on tool usability and functional coverage, with the objective to highlight potential weaknesses and to provide design recommendations useful while implementing subsequent versions of the TERESA tool.

The experiment consisted in starting with a given task model created with CTTE 1.5.7 and obtaining the concrete user interface for both desktop and mobile phone using the version 1.1 of TERESA tool. The exercise goal was to realize a simple version of an e-desk application allowing three main actions: the registration to the service by inserting a username and a password, the selection of a location (workplace, home, travel or vacation), and the selection of an application from a menu. The applications offered are different in the desktop and in the mobile versions of the service.

The actions to be performed, such as Generate Enabled Task Sets, Generate Abstract User Interface, etc. were predefined in order to require the access to every tool menu. For each step evaluators were asked to record any difficulties they may have encountered in achieving the goal and their suggestions to improve the user interface. In addition, they were invited to provide comments about: approach, functionalities and result produced, reporting advantages/disadvantages with respect to traditional methods and providing indications on additional functionalities they would like to introduce.

The first evaluation resulted in an amount of data about the aspects considered. The analysis has been conducted in two steps. Firstly the raw comments have been abstracted to recurrent issues aggregated with a functional criterion, counting the occurrences of each issue; this step has been conducted iteratively, in order to progressively obtain a clean taxonomy. In the second step the taxonomy has been presented to the evaluators, who were requested to express for each issue a relevance assessment (high, medium, or low) a sort of quantitative measurement of the 'severity' of the problem; from such new data a relevance index has been synthesized for each item.

Table 2 reports as an example the analysis of the results related to the Final UI generation functionality. Seven main aspects requiring attention were identified.

Issue	***Occurrences***	***Relevance***
Final UI generation	*5/5*	
Messages language	1/5	low
Inserted data not reported	3/5	high
Destination folder	2/5	high
Windows unresizable and overlapping	3/5	medium
Not intuitive fields and controls	3/5	high
Not intuitive presentations structure and content	1/5	high
Relation operator for mobile UI	2/5	high
No browse button for URL	1/5	medium
Window consistence	1/5	low
Confirmation panel	2/5	high

Table 2: Example of analysis results

The results of analysis have been reported to the development group, which integrated them in the new version of TERESA used for the second experiment.

The next version of TERESA was substantially improved with respect to the first prototype taking into account the results of the experiment. For example the effect of the heuristics used for combining two or more PTS has been made more predictable, the AUI generation window has been redesigned in order to be intuitive and usable, the Final User Interface Generation has been improved by the introduction of a preview windows, the task corresponding to an object can be automatically identified, and some model editing options have been introduced.

A second experiment has then been conducted in order to collect more information about developer satisfaction and cost/effectiveness of the approach.

The subjects involved in the second experiment were the same people that performed the first experiment, so they could better appreciate the modifications introduced in the tool and provide specific feedback.

The experiment consisted in developing a prototype version of an e-Agenda application running on both desktop and mobile phone and including the following functionalities: visualization of the appointments of a single day; visualization of the details of each appointment; possibility of inserting/modifying/deleting an appointment. This had to be realized in two ways:

- At first using traditional techniques such as a template for the design phase and Microsoft Front Page or Netscape Composer for the implementation phase.

- Then using tool-supported techniques: CTTE 1.5.7 for task tree realization and version 1.5 of TERESA tool (updated taking into account the results of the first experiment) for XHTML and XHTML Mobile Profile pages generation.

Every evaluator had been asked to perform the same task using the two approaches, in the order specified above.

The evaluators have been required to collect quantitative metrics related to development efficiency, such as the total effort needed to complete the exercise expressed as creation or rework time and categorized by process phase, as well as the number of errors introduced. Moreover, they have been required to express their judgment on specific TERESA characteristics such as support offered to identify the most suitable interaction techniques, support offered to compose interactors in the interface, and others aspects related to developer satisfaction and product maintainability/portability by a rating from 1 (poor) to 5 (very good). In case of negative evaluation they were invited to provide an explanation note and suggestions for improvement.

The results of the second experiment show how developers' productivity is affected by the use of the tool. Data about time performance have been collected in each phase of the experiment and summarized through average values.

Results graphically illustrated in Figure 2, show similar total times for the traditional and TERESA approaches, with different distributions over the development phases and between first version and rework time.

The tool-supported methodology offers a very good support to fast prototyping, producing a first version of the interface in a significantly shorter time.

This difference is significant and interesting, considering that developers are often required to implement in a short time different interface prototypes to present and discuss with their customers, and then to refine later the selected version.

On the other side rework time results increased. In particular the design phase results negatively affected while the development phase is positively impacted by the use of the semi-automatic environment.

This is mainly due to the greater familiarity of the subjects with traditional techniques than with model-based techniques and notations. Future refinements to TERESA and a continuous use of the tool in the software production process are then expected to consistently reduce rework time needed and to confirm the advantages of the proposed tool supported methodology.

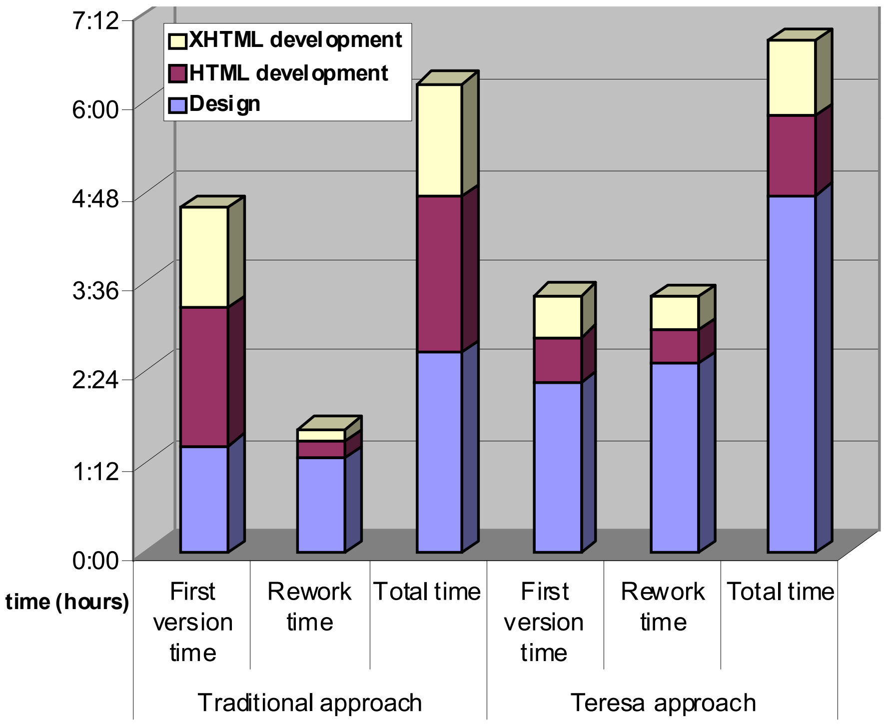


Fig. 2: Comparative results on time performance.

Even more interesting than the time performance itself have been the comments of the evaluators, who remarked an increased design overall quality and appreciated the benefits of a formal process supporting the individuation of the most suitable interaction techniques. For example, the subjects reported satisfaction about how the tool supported the realization of a coherent page layout and identification of links between pages; they noticed and appreciated the improved structure of the presentations and more consistent look of the pages resulting from the model-based approach, as well as the reduced risk to forget the formal specifications; they pointed out an increased consistence between desktop and mobile version.

5. Conclusions and Acknowledgements

In this paper a model-based approach for designing and developing multi-platform applications has been presented and discussed through an experimental evaluation.

In summary, TERESA emerged from the evaluation as an appealing and promising solution for designing and developing UIs on multiple and heterogeneous devices.

At the same time the evaluation methodology and criteria we introduced appears to be general and applicable to different systems.

Further activities will include additional experiments focusing on the final product and involving end users.

The TERESA tool is publicly available at http://giove.cnuce.cnr.it/teresa.html.

This work has been supported by the IST V Framework CAMELEON (Context Aware Modelling for Enabling and Leveraging Effective interaction) project. More information is available at http://giove.cnuce.cnr.it/cameleon.html.

We also would like to thank the colleagues Cristina Barbero, Simone Martini, Bianca Russillo and Massimiliano Fliri for participating to the experimental evaluation and for the useful discussions.

6. References

Abrams, M., Phanouriou, C., Batongbacal, A., Williams, S. & Shuster, J. (1999) *UIML: An Appliance-Independent XML User Interface Language*. Proceedings of the 8th WWW conference, Toronto, Canada.

Berti, S., Mori, G., Paganelli, L., Paternò, F., Santoro, C., Calvary, G., Coutaz, J., Thevenin, D., Vanderdonckt, J. & Bouillon, L., (2003) *Tools for Model-Based Design of Multi-Context Applications*. Deliverable D2.1, CAMELEON project.

Berti, S. & Paternò, F., (2003) *Model-based Design of Speech Interfaces*. Proceedings of DSV-IS 2003, Funchal, Madeira, Springer Verlag.

Beyer, H. & Holtzblatt, K. (1998) *Contextual Design: Defining Customer Centred Systems*. Morgan Kaufmann, San Francisco.

Calvary, G., Coutaz, J. & Thevenin, D. (2001) *A Unifying Reference Framework for the Development of Plastic User Interfaces*. Proceedings of EHCI 2001, Toronto, Canada, Springer Verlag.

Chesta, C. & Fliri, M., (2003) *Early e-Desk Prototype Description*. Deliverable D3.3, CAMELEON project.

Chesta, C., Fliri, M., Martini, S., Russillo, B., Barbero, C. & Raymond, S., (2003) *First Evaluation of Tools and Methods*. Deliverable D3.4, CAMELEON project.

Coutaz, J., Balme, L., Barralon, N., Calvary, G., Demeure, A., Lachenal, C., Rey, G., Bandelloni, R. & Paternò, F., (2003) *Initial Version of the CAMELEON Run Time Infrastructure for User Interface Adaptation*. Deliverable D2.2, CAMELEON project.

Einsenstein, J., Vanderdonckt, J. & Puerta, A. (2001) *Applying Model-Based Techniques to the Development of UIs for Mobile Computers*. Proceedings of IUI'01, Santa Fe, New Mexico, ACM Press.

ISO9241-11 (1991) *Ergonomic requirement for office works with VDT's – guidance on usability*. Technical report, International Standard Organisation.

Limbourg, Q., Bouillon, L., Vanderdonckt, J., Michotte, B., Santoro, C. & Paternò, F., (2004) *CameleonXML V1.0 Description Document*. Deliverable 1.3, CAMELEON project.

Mori, G., Paternò, F. & Santoro, C. (2003) *Tool Support for Designing Nomadic Applications*. Proceedings of IUI'03, Miami, Florida.

Myers, B., Hudson, S. & Pausch, R. (2000) *Past, Present, Future of User Interface Tools.* Transactions on Computer-Human Interaction, ACM Press, 7(1), 3-28.

Nielsen, J. (1994) *Usability Engineering.* Morgan Kaufmann, San Francisco.

Paternò, F. & Leonardi, A., (1994) *A Semantics-based Approach to the Design and Implementation of Interaction Objects*. Computer Graphics Forum, Blackwell Publisher, 13(3), 195-204.

Paternò F., (1999) *Model-based design and evaluation of interactive applications*. Springer Verlag, ISBN 1-85233-155-0.

Paternò, F. & Santoro, C., (2003) *A Unified Method for Designing Interactive Systems Adaptable to Mobile and Stationary Platforms.* Interacting with Computers, Elsevier, 15(3), 347-364.

Puerta, A.R. (1997) *A Model-Based Interface Development Environment*. IEEE Software, 14(4), 40-47.

Puerta, A.R. & Eisenstein, J. (2002) *XIML: A Common Representation for Interaction Data*. Proceedings of IUI'02, San Francisco, California, 214-215.

Rich, C. & Sidner, C. (1998) *COLLAGEN: A collaboration manager for software interface agents*. User Modelling and User-Adapted Interaction, 8(3/4), 315-350.

Schneiderman, S. (1998) *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison Wesley, Reading, MA.

Thevenin, D. & Coutaz, J. (1999) *Plasticity of User Interfaces: Framework and Research Agenda.* Proceedings of Interact'99, Edinburgh, Scotland, 110-117.

PsychNology Journal, 2004
Volume 2, Number 1, 140 - 162

User Interaction with an Automated Solver The Case of a Mission Planner

Gabriella Cortellessa, Amedeo Cesta[♣], Angelo Oddi and Nicola Policella

ISTC-CNR
National Research Council of Italy
Rome, Italy

ABSTRACT

An effective interaction with the user is a key aspect for the success of technological tools applied to both everyday and highly specialized tasks. This paper shows features of MEXAR, an intelligent system that solves a mission planning problem related to the MARS EXPRESS program of the European Space Agency. The paper describes the MEXAR interaction module developed to support human mission planners in a specific daily task, which consists in generating commands for downloading the on-board memory of the spacecraft. The interactive environment of MEXAR helps a user to analyze the current problem and takes planning decisions as a result of an interactive process enhanced by various elaborate facilities. Different interactive techniques have been integrated to address two different aspects: (a) developing trust on behalf of the user in the automated algorithms; (b) promoting a deep participation of the user during problem solving. An integral part of the tool development process has been a usability study on MEXAR'S Interaction Module, aimed at discovering possible problems in user-system interaction. This paper discusses how the enhancement of both transparency and usability of automated decision making tools is fundamental for users' acceptance of artificial support systems and their profitable deployment in real world applications.

Keywords: *human-computer interaction, interactive problem solving, planning and scheduling problems, user involvement, space applications.*

Received 20 January 2004; received in revised form 5 April 2004; accepted 6 April 2004.

1. Introduction

State of the art interactive technology could be very useful in supporting human tasks, and its application may vary from simple systems able to support daily human activities, to more complex and sophisticated decision support tools (e.g., in the context of medical environment, transportation domain, space missions) that help human decision makers in taking important and difficult decisions. Unfortunately, very often,

[♣] Corresponding author:
Amedeo Cesta,
ISTC-CNR
Viale Marx 15, I-00137 Rome, Italy
Phone: +39-06-86090-209 fax: +39-06-82473
E-mail: a.cesta@istc.cnr.it

the automated tools present deficiencies and shortcomings resulting in a non effective usage or in the user not to trust them. There are at least a couple of reasons for this mistrust of the automated systems. Most of interactive devices are endowed with a “bad designed” interface, which does not present information properly so to satisfy the user's needs. A user tends to be skeptical toward the use of a *black box* that hardly explains choices, actions and results. An appropriate and effective representation of problem modelling, all objects and entities involved and the solutions or the advices proposed by the artificial aid are to be guaranteed, in order to promote a more fluid and flexible interaction.

The naturally conservative behaviour of people in changing their habits makes it difficult the spread of such a supporting technology, in particular in those contexts in which critical decisions are to be taken. The final user of systems of this kind is used to make decisive choices and perform complex tasks completely “by hand” and the attempt to *automate decisions* or *jobs* it is not a trivial problem. Users tend not to abandon their traditional way of working and get into new habits, unless, we believe, this entails higher quality, higher speed in obtaining outcomes, more stimuli, less annoyance, certainty of the correctness of the results and above all the possibility to actively participate being in charge of the final decisions.

For these reasons the design of interactive systems is a hard and challenging problem and the success of their use is also dependent on an effective and useful *interaction module*. It is worth noting though that most of the automated systems belonging to the past generation, failed in their attempt to support users' tasks or problem solving due to a common lack of attention to the user. We are referring to a usual attitude of focusing on the development of powerful and efficient algorithms to automate tasks, while ignoring the important issue of making systems usable and interactive. Most of the automated systems do not support users properly because they are lacking an effective interaction module that keeps the user in the loop. The interaction environment should guarantee a friendly and comprehensible representation of the problem, the problem solving process and the solutions, especially in those cases in which the automated system is devoted to support a user in solving complex problems. It should allow a user to verify the correctness of the results, the possibility to express her/his own preferences, and a continuum in changing her/his way of working, by allowing a gradual adaptation to the innovation. The interaction module bridges the gap between users and artificial solver hiding useless details and technical complexity. It should promote a stimulating and effective work environment to enhance human problem solving capabilities.

In this paper we present our experience in developing a tool for supporting the user in the context of a space mission. We report on work done to design and realize an intelligent system able to represent in a compact and meaningful way a huge amount of complex information and solve an involved problem. The context of our study is MARS EXPRESS, a space probe launched by the European Space Agency (ESA) on June 2, 2003 that has been orbiting around Mars since the beginning of 2004 for two years.

The paper briefly describes the addressed problem and the problem solving, then mainly focuses on the description of the interactive techniques designed and implemented within an integrated system, named MEXAR.

The paper is organized as follows: Section 2 describes the MARS EXPRESS mission and briefly introduces the addressed sub-problem (MEX-MDP problem); Section 3 shows the software architecture designed to solve the examined problem; in Section 4 the main ideas to design interactive services supporting human planners are introduced, together with a description of their implementation within MEXAR; Section 5 shows how usability techniques have been applied in MEXAR interface. Section 6 discusses some related work, while some conclusions on our experience end the paper.

2. A study in the context of the space mission MARS EXPRESS

An orbiting spacecraft continuously produces a large amount of data which derives from the activities of its scientific instruments (payloads) and from on-board device monitoring and verification tasks (the so called *housekeeping data* analyzed to check the safety of the spacecraft). All this data, usually referred to as *telemetry*, is to be transferred to Earth during downlink connections. MARS EXPRESS is endowed with a single pointing system, thus during regular operations, it will either point to Mars and perform payload operations (scientific observations) or point to Earth and transmit data through the downlink channel. As a consequence on-board data is first stored on the on-board memory then transferred to Earth during *temporal visibility windows*. MARS EXPRESS contains seven different scientific payloads which will gather different data on both surface and atmosphere of the Red Planet. During the operational phase around Mars a team of people, the Mission Planners, are responsible for deciding the on board operations of MARS EXPRESS. Any single operation of a payload, named POR for Payload Operation Request, is decided well in advance through a negotiation phase among the different actors involved in the process (e.g., scientists, mission planners, flight dynamics experts). This negotiation causes (a) acceptance or rejection of a single POR, (b) assignment of a start time to the accepted PORs. The mission planners' goal consists of guaranteeing an acceptable *turn-over time* [3] from the end of the execution of the POR to the availability on Earth of data generated by that POR.

An effective management of on-board memory and a good policy for *downlinking* [4] its data are very important for a successful operation of the spacecraft. The authors' study has addressed the problem of automatically generating downlink commands for on-board memory dumping. They have formalized the problem as the MEX-MDP (MARS EXPRESS Memory Dumping Problem), defined a set of algorithms to solve the problem, and implemented an interactive system, named MEXAR, which allows human planners to continuously model new MEX-MDP instances, solve them, inspect a number of problem and solution's features. The reminder of this section briefly reports the solved problem while a more detailed description is given in (Cesta et al., 2002).

Fig. 1: The mission of MARS EXPRESS is to orbit Mars and provide scientists with high definition images and spectra. An artist's impression of MARS EXPRESS in orbit around Mars is provided on the left, while the picture of the right shows one the first images of the Martian surface taken by MARS EXPRESS – Both pictures are courtesy of ESA.

The basic ontological objects that describe the MEX-MDP problem are *resources* or *activities*. *Resources* represent domain subsystems able to give services (e.g. photo-cameras, on-board memory); *activities* model tasks to be executed (e.g., take picture)

[3] With *turn-over time* we indicate the temporal delay between the end of a scientific observation and the instant in which the related data produced by the observation is completely transmitted to Earth.
[4] The term *downlinking* will be used in the paper to indicate the activity of data transmission from the satellite on-board memory to the ground station on Earth.

using resources over time. A set of *constraints* defines needed relationships between the two types of objects (e.g., the on-board memory has a limited capacity).
The relevant types of resources in MEX-MDP are the set of packet stores, the set of on-board payloads and the set of communication channels:

- *Packet Stores.* The on-board memory (Solid State Mass Memory) is subdivided into a set of independent packet stores that cannot exchange data. Each one has fixed capacity and an associated priority value reflecting the importance of the stored data. Data produced by scientific observations is organized in data packets and then stored in the assigned packet store.
- *On-Board Payloads.* An on-board payload represents a scientific instrument used to perform observations. Data produced by scientific observations is organized in data packets and then stored in the assigned packet store.
- *Communication Channels.* These resources are characterized by a set of separated communication windows identifying intervals of time for downlink (*temporal visibility windows*).

The amount of each resource is constant and known in advance.
Activities describe *how* resources can be used. Two types are relevant in MEX-MDP: payload operations and memory dumps. Each type of activity is characterized by a particular set of resource requirements and constraints.

- *Payload Operations.* A payload operation request (POR) corresponds to a scientific observation. Each observation generates an amount of data that, according to the MARS EXPRESS operational modalities, is decomposed into different *store* operations, and distributed over the set of available packet stores. Payload Operations represent the *input* for the system.

- *Memory Dumps.* A memory dump operation transfers a set of data from a packet store to a transfer device. These activities represent data transmission through the communication channel. Memory Dumps represent the *output* of the system.

Given a set of POR operations, a *solution* to a MEX-MDP problem is a set of Memory Dumps that satisfy all the constraints imposed by the system (finite capacity of the packet store, visibility windows, etc.). An additional goal is to find *high quality solutions* with respect to a set of evaluation parameters; usually high quality solutions deliver all the stored according to a definite policy or objective function (e.g., data delivery time to Earth).

3. An intelligent software for scheduling data return

In order to solve MEX-MDP problems we designed the MEXAR architecture which uses Artificial Intelligence (AI) techniques to model and solve the problem. In particular our system combines Planning and Scheduling (P&S) technology with flexible interaction modalities in an overall interactive tool. It is worth noting that P&S technology is mainly devoted to *automate decisions* while common practice in space missions is that *decisions are taken by humans* and only low level activities are automated. The traditional management of the data return problem is an example of a mostly "hand made" activity where the human mission planner works at a very low level of abstraction, consulting different support tools only for specific computation. Moreover, as already said in the introduction, users tend to be skeptical toward novelty and changes in their way of work, especially in those contexts where difficult decisions are to be taken that may cause undesired failures. For these reasons we decided to use a conservative approach to the problem, and designed a decision support aid that benefits from the computational strength of the automated algorithm but leaves the humans in charge of their responsibilities. The general framework of the system is composed of two modules as it is shown in Figure 2.

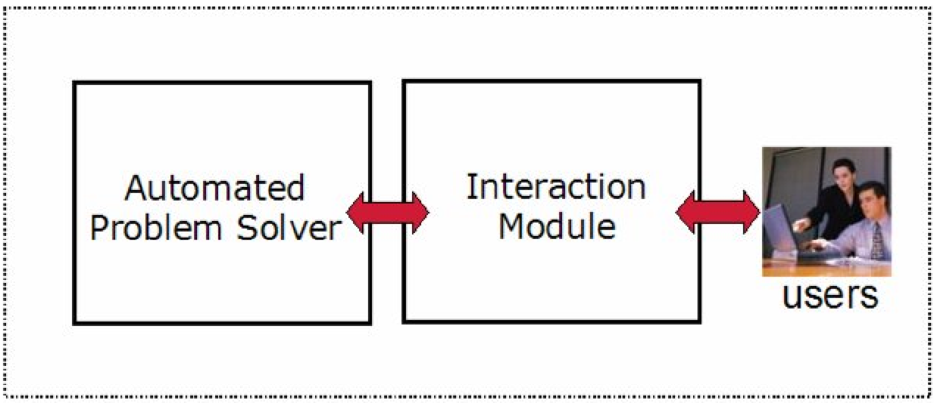


Fig. 2: Generic architecture for an interactive intelligent system

The first module, named *Automated Problem Solver*, models the real domain, and captures the dynamic rules according to which the domain evolves. This module relies on an internal representation of the problem which is suitable for both automated algorithms and interaction with the users. The internal representation models relevant knowledge as a set of constraints and provides data structures for representing the domain, the problem, the solution and its management. On top of this representation a set of automatic algorithms guide the search and generate problem solutions. The second component of the system is the *Interaction Module* that directly interacts with the user, and allows her/him to take part in the process of finding a solution by providing advanced interactive facilities. It represents the communication channel between the user and the automated solver and a means to exploit powerful features of the automated system. Aspects of the two modules are described in the rest of the paper.

3.1 Problem Solving in MEXAR

Our approach to solve MEX-MDPs is grounded on the Constraint Satisfaction Problem (CSP) paradigm (Tsang, 1993). This problem consists of a set of variables $\{v_1,v_2,\ldots,v_n\}$, each with its own domain of values $\{D_1,D_2,\ldots,D_n\}$, and a set of constraints $\{c_1,c_2,\ldots,c_m\}$, that define the relations among those possible values. A solution *s*, consists in assigning to each variable one of its possible values, $v_i = d_i$ with d_i belonging to D_i, such that all the constraints are satisfied.

A MEX-MDP instance has been represented as a Constraint Optimization Problem (COP), the optimization version of a CSP. In this case the assignment that optimizes a defined objective function *f(s)*, is selected. In our case the objective function concerns the temporal delay between the end of a scientific observation and the instant in which all the related data is completely downlinked to the ground (*turn-over time*). For further details, we defer the reader to (Oddi et al., 2003) where a thorough discussion for both MEX-MDP representation and the automated algorithms is provided. In this context we highlight our attempt to develop a support tool that helps the human mission planner by providing advanced problem solving facilities, without changing radically her/his traditional work modality.

On top of the constraint based representation for MEX-MDP, a multi-strategy solving method with a portfolio of algorithms has been developed. The idea is to endow the human mission planner with a set of solving algorithms she/he can choose among, in order to automatically get a solution to the problem. In particular MEXAR provides two solving algorithms, a greedy solver and a random sampler, and an optimization procedure based on tabu-search to iteratively improve solutions (Oddi et al., 2003). In this light the problem solving process can be seen as a two step procedure: first, an initial solution is found by using either the greedy algorithm or the random sampler. Once an initial solution is obtained, it is possible to look for improvements by using an optimization strategy. The optimization method explores the solution space according to different principles trying to achieve a solution with a better value of the objective function. A “friendly” graphic interface abstracts the complexity of the problem representation and the technicality of the solving algorithms and provides a high level description the algorithms' features. Through this representation a user can easily choose and configure the solving algorithm by tuning the requested parameters according to her/his preferences. In this way she/he maintains a level of control on the solving process.

4. The user interaction with MEXAR

MEXAR is endowed with a sophisticated interaction module that allows a user to easily supervise and control the entities of the domain and the whole solving process, being aware of all the steps the solver goes through. Specialized interactive functionalities provide a user with satisfactory information services and control facilities. They put at user's disposal the possibility to inspect the problem and obtain an initial solution by choosing among different solving algorithms she/he can personalize. Moreover an advanced environment allows an expert user to select the best solution for the execution as result of a “step by step” procedure enhanced by incremental improvement algorithms, evaluation services, and graphic comparison functionalities.

4.1 Designing interactive functionalities

Before describing the interaction functionalities MEXAR is endowed with, we shortly introduce two main problems that have been taken into account while developing the Interaction Module. In our opinion, such problems need to be constantly addressed in the design of interactive systems.

- **Visualization problems.** It is the need to guarantee a certain level of “transparency” to the user, providing comprehensible and significant representations of the real domain, the problem, its solutions and the problem solving process. These needs are also related to the mentioned general skeptical attitude of the users toward automated systems. In order to gain user's trust an automated system should be endowed with clear and expressive representation services. We refer to this as to the *glass box* principle, contrasting the widespread trend, among the users, to consider the automated systems as a *black box* to be distrustful or suspicious of.

– **Interactive user participation.** The idea is to capture different skills that a user and an automated system can apply to the resolution process. Typically an algorithm can perform better on conducting repetitive search steps that are not possible for a human user, while the user usually has more specific knowledge on a domain that is often difficult to formalize in terms useable by an algorithm. The overall systems Human Planner/Artificial Solver could be considered as more powerful and able to more efficiently solve a problem. The Interaction Module plays a crucial role in enabling such cooperation and should provide interactive services and functionalities to promote a combined problem solving.

In the following we provide a description of how these ideas have been implemented within the system in the attempt of addressing the two main issues discussed above.

4.2 Implementation of the functionalities

While designing an Interaction Module for MEXAR we have been thinking of how to solve the problem of visualizing a huge amount of complex information, providing a user with an environment that enables to express her/his preferences and actively participate in the problem solving.

Visualization problem

The CSP approach underlying the automated solver uses a symbolic model of the domain features subdivided between activities and resources. Based on a *glass box* principle, the MEXAR interface provides a meaningful visual representation for all the entities relevant in this domain model.

A specific dialogue allows choosing one of the problems to be solved and to instantiate its CSP representation. Such representation is used to display information of the interaction panel. Figure 3 presents the basic layout of the MEXAR interface. In particular it shows the basic visualization of a MEX-MDP problem.

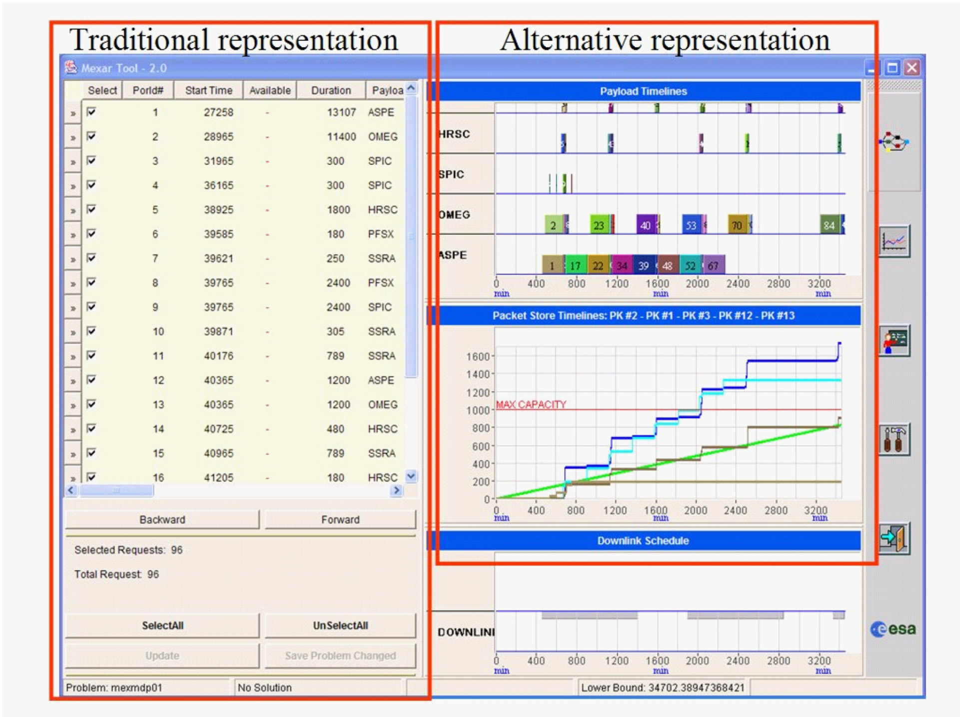


Fig.3: Visualization in MEXAR: examining problem features

The left panel in the layout shows the list of Payload Operation Requests (PORs) in textual form, that is a detailed list of information on the input activities, their temporal allocation, and the distribution of their data on different packet stores. Then three different panels on the right part of the layout (one on top of the other) represent the timelines (distribution over time axis) of the different domain features relevant for the user. The first panel (higher panel on the right) shows the Gantt Chart of PORs in the problem. This is a graphic representation of the input activities of the problem over time. For each payload there is a different timeline where each POR is represented by a coloured rectangle labelled with a natural number (PorId#) starting from its start time.

A graphic representation of the temporal function representing the volume of data stored over time is given in the central panel on the right. A line labelled "MAX CAPACITY" indicates the packet store capacity. As the reader can notice, data in the packet stores in Figure 3 will exceed the max capacity which entails that downlinking is necessary to avoid overwriting[5].

The last panel on the right will contain the memory dump activities (solution), after the solver activity (see Figure 4).

---

[5] Each packet store is managed cyclically. When a packet store is full, any new information overwrites the older one.

These three panels form the basic information the user should be familiar with in order to develop trust in what the solver is representing and managing. A major software development effort has been needed by those three panels that, for example, should be constantly synchronized in their scrolling to represent a consistent model of the world. This temporal representation of the internal symbolic model used by the CSP solver has been instrumental for convincing the ESA personnel that the solver was addressing exactly the problem they had.

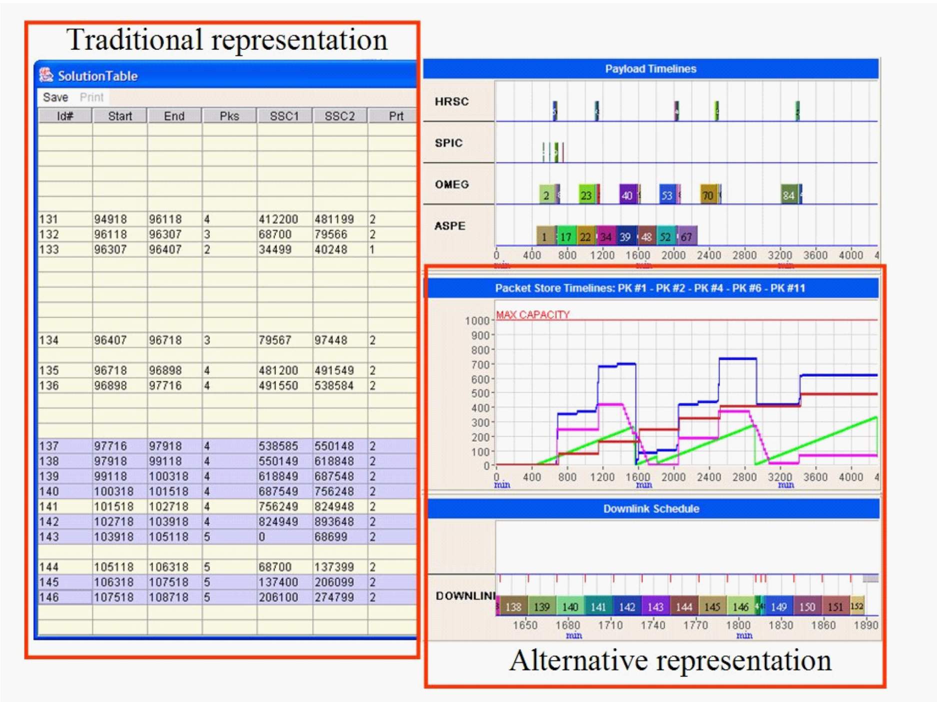


Fig. 4: Visualization in MEXAR: studying a solution

In Figure 4 a solution for the problem is shown. In the left part of the panel a textual representation of the solution is given, while an alternative graphic representation is shown in the third panel on the right.

Before solution (see Figure 3), the user can see static features of the problem, and inspect any of its aspects in detail. After solution, the user can immediately understand if a solution has been found, and have a visual view of the load on each packet store (see the difference between before and after on the center right panel). In the example of Figure 4 the maximum capacity constraint is never violated due to the effect of different dumps (the packet stores profile is always below max capacity). The sequence of dumps on the communication channel (lower right panel) completes the information on the solution. In this last panel a bar represents the time intervals where it is possible to perform memory dumps (visibility windows). Intervals where dumping is not possible are drawn in grey.

In providing different and alternative representations within MEXAR we paid attention to design an interaction modality quite close to the traditional way of working of the human planner. In this way a user can count on a system that facilitates her/his task by providing information and solutions close to the way she/he is used to, and get gradually acquainted with the alternative interaction modalities. The “traditional representation” of the problem and the solutions provides a detailed description very close to the one the human mission planner is used to. We are referring to the PORs list in textual form, specifying in detail the related information (Figure 3) and to the solution table that reconstructs all the details concerning the solution of the current problem (Figure 4). The alternative graphic view represents a more compact, intuitive and high level vision of the problem and the solution. It is worth saying that the solution table reflects the current way of working at ESA, in fact mission planners mainly deal with numerical data contained in spreadsheet tables. Using the table it is possible to check for example (a) how the data from a single POR is segmented in different dump operations, (b) how the time of data return has been generated, etc. In general, it could be also possible to directly generate the dump commands from the lines of the table. In fact, the whole table can be saved as a separate file and manipulated by different programs.

A user is free to use only one or both modalities. The two representations are indeed linked and synchronized to each other through a set of interactive links. For example, the alternative representations for the PORs (the one on the table and the one on the timelines) are not redundant as they focus on different aspects. In fact, the list of PORs on the left gives detailed information on the payload activity, while the Gantt on the right focuses on their temporal allocation, and allows to have a feeling of the impact of any POR on the packet stores *via* the other synchronized right panels.

Another feature of the module is the possibility to evaluate a solution according to some metrics. This possibility enables the user to easily estimate the quality of a solution with respect to some chosen parameters. In fact, a graphic evaluation has been added to obtain an immediate level of evaluation of the current solution.

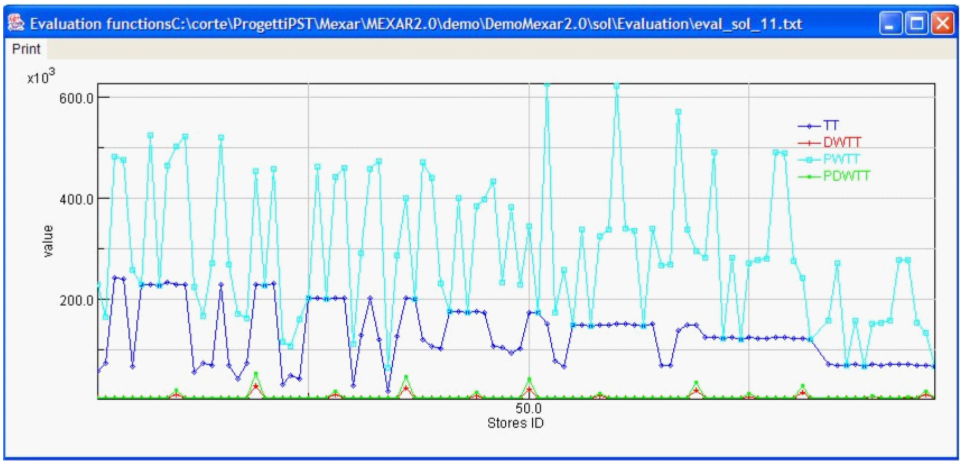


Fig. 5: Evaluating a solution.

When calling the command from the menu a new dialogue window allows choosing one or more evaluation functions. Figure 5 shows an example, where the x axis represents the PORs, while the y axis represents the turn-over time.

A number of other small interaction features are implemented as basic functionalities but are not described here because they do not change the general perception of the interaction flow.

User participation in problem solving

Once the human planner has deeper knowledge of the problem and all the aspects it involves, she/he can start a different level of interaction with the system trying to contribute with her/his expertise and judgment to the problem solving activity. In this way she/he can possibly choose either to completely entrust the system with the task of finding a solution or to participate more interactively in the problem solving process.

MEXAR puts at her/his disposal a second interaction layout, called Solution Explorer that is intended as an example of such an enhanced interaction environment. This second layout has been created mostly for showing ESA personnel an example of advanced functionality based on our interactive problem solving technology. As said before the Automated Solver allows a user to apply different solving methods to the same problem. Specific functionalities enable the user to save different solutions for the same problem and to guide a search for improvements of the current best result by applying different optimization algorithms. The idea behind this aspect of MEXAR is to deeply involve the expert user in the problem solving process. A user might generate an initial solution, save it, try to improve it by local search, save the results, try to improve it by local search with different tuning parameters and so on.

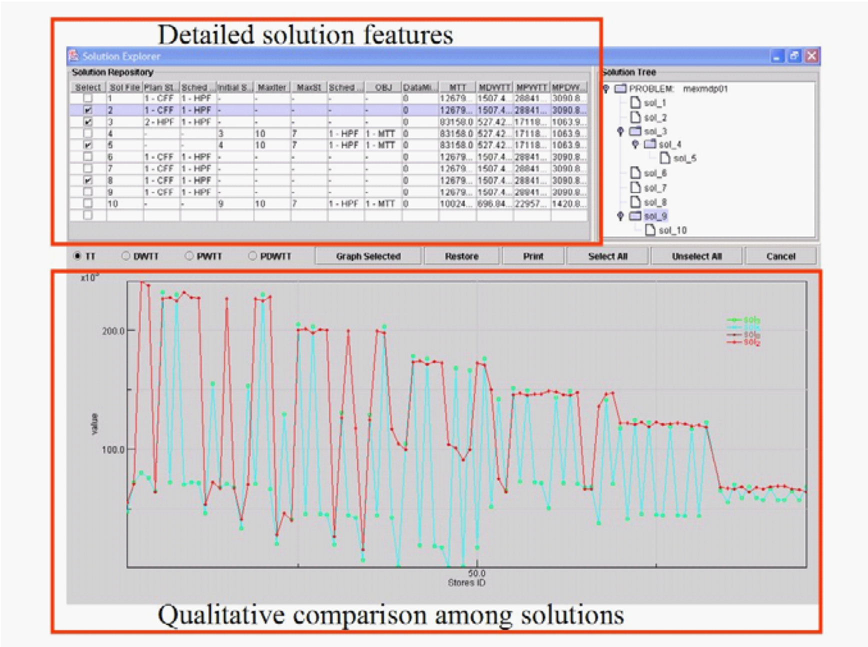


Fig. 6: Involving the user in the problems solving process. A step in exploring the solutions space.

This procedure can be repeated for different starting points, resulting in the generation of different paths in the search space. Using both the evaluation capability on a single solution and her/his own experience the user can visit the different solution series, all of them saved, and, at the end, choose the best candidate for execution. Figures 6 and 7 show two examples of use of the interaction environment, in particular depicting a single problem at different stages of exploration. Studying the examples, it is possible to see that our idea has been again that of facilitating the analysis of the current problem by providing multiple representations of the solutions features. A user has different tools to evaluate the solutions and can either generate new ones or choose the best one according to different temporary criteria.

The ideas behind the iterative construction of a solution in MEXAR provides a concept of human guided search (see also different approaches like (Anderson et al., 2000)), that can be very useful in the search for a solution to complex problems. Indeed, in the current version, the Solution Explorer provides an inspectable repository.

After selecting a solution from the tree, the user can inspect various quality measures. Attention has been devoted to building an interface that allows the user to be in control on the different solution paths without getting lost in the increasing number of solutions. This environment can become increasingly useful if the user were endowed with more complex and powerful means to influence the artificial solver.

Fig. 7: Involving the user in the problems solving process. A more advanced state in the exploration.

Somehow, our desiderata would be to continue the development of the system in this direction, which is not part of the initial study, in order to obtain a more advanced example of mixed-initiative interaction (Burstein & Mc-Dermott, 1996; Cohen et al., 1999). This term generically identify the problem of creating solving environments in which users and automated systems cooperate in problem solving contributing with their respective abilities. Nevertheless the Solution Explorer represents a proof of concept which is very useful for tuning future directions.

5. Evaluation

MEXAR has been delivered to ESA-ESOC in May 2002 and is currently available to the mission planners. Users' reactions to the system have been quite positive. We highlight in particular a real interest in the idea of using an automated tool that performs boring and repetitive tasks on users behalf, while preserving humans control on the flow of actions and the possibility to choose the final solution supported by the automated tool.

Interactive systems represent a very interesting attempt to benefit from complementary reasoning styles and computational strengths of both human and artificial solvers. However, because of their composite nature, the design and implementation of such mixed and integrated systems is an arduous and stimulating challenge, likewise for the measurement of their effectiveness and utility. The diversity and complexity of the two involved entities, the human user with her/his unpredictable and sophisticated reasoning and the artificial machine, with its computational complexity and technicality, together with the uncontrollability and uncertainty of the environment, makes it difficult to design precise and effective evaluation methodologies. For these reasons we believe that it is worth investing in the study and design of evaluation techniques that take both the human and the artificial component into account to measure the effectiveness and quality of the overall system. A valid possibility is to exploit existing methodologies from close or related research fields.

In our case, given that our main concern in the development of the system was making it useful for the user, we decided as a first step to conduct a usability study on the Interaction Module. In fact, as already mentioned, one of the first problems we noticed when started the study was users' skepticism. A system difficult to use or with incomprehensible features may increase the inertia with which users get used to novelty and appreciate its benefit. A usability study allowed us to detect existing problems and propose possible solutions to them. In the following a report on this preliminary study is given together with a description of the experiment designed to improve the user-system interaction in MEXAR.

5.1 Methodology

The usability evaluation of MEXAR interface has been conducted by using the "*Think Aloud"* observational technique. The essence of this evaluation technique, which can be considered the best discount usability engineering method (Nielsen, 1993), consists of asking the users to verbalize their thoughts while performing certain tasks and interacting with the system. The experimenter observes silently the interaction session, and records user's actions and thoughts, focusing on the difficulties and problems encountered. A subsequent data analysis phase allows both to identify the positive and negative features of the interface and to propose methods that would solve the identified problems. This technique highlights usability problems in terms of difference between the interaction logic of the system and the model adopted by the user to perform certain actions.

Participants

Four participants took part in our usability experiment. During the development of MEXAR it was not possible to perform experiments directly with real users. As a consequence we performed this usability study with people unaffiliated with the project but having experience in scheduling and planning systems. In particular, two of them were PhD students and two were Research Programmers. Three out of four received training prior to their test, while the forth tester did not receive any training.

Apparatus

The experiment has been conducted on a Personal Computer Pentium 4 - 1,70 GHz - 256 MB RAM under Win XP. The software had been previously tested through a simulation session. During the experiment the interaction sessions have been recorded by using a voice recorder and a screen capture software to register user's thoughts and actions.

Tasks and Procedure

Tasks have been subdivided according to MEX-MDP management phases and designed to completely test system functionalities and services. Three different subtasks have been selected:

- MEX-MDP *Problem Analysis*. Participants have been asked to load an instance of a MEX-MDP problem and inspect problem features (e.g. difficulty, data volume produced by PORs, etc.)
- *Solve* MEX-MDP *problem*. Participants have been asked to find a solution to MEX-MDP instances using the Automated Problem Solver and analyze solution features (e.g. number of Memory Dumps, solution quality, etc.)
- *Select final solution to* MEX-MDP *problem*. Participants have been asked to select a solution considered the best candidate for the execution. They could study the current solution and improve it by using the local search procedure or find completely new solutions through the Solution Explorer. Eventually they had to indicate the final solution considered the best one according to certain quality parameters and their own judgment.

Two training sessions preceded the experiment. They allowed users to exercise the actions needed to carry out the required tasks. Participants received written instructions on tasks and how to carry them out. The experimenter integrated the tasks description by answering additional questions to clarify user's doubts. Training sessions lasted respectively 60 and 30 minutes. The day after the training sessions a detailed description of the tasks has been presented to the users and subsequently they have been asked to carry out each subtask. The experiment lasted about 60 minutes. At the end of the experiment a further interview has been conducted to discover further problems and take note of additional users' advice. Finally the recordings have been analyzed and experiment results have been written in the form of Usability Aspect Reports (UARs).

5.2 Results and Discussion

The final step of the experiment consisted in analyzing the UARs to find out interaction problems and conceive possible solutions. For each problem we thought of a solution aimed at filling the discrepancy between our implementation of the system and the model that the user had of the particular involved aspect. In fact, the proposed solutions derived from a reasoning process devoted to more specifically support user's goal while she/he was carrying out required tasks. A further step allowed us not only to perform a number of local improvements on features of the interactive services, but also to formulate some remarks on the interactive aspects we consider relevant to the design of an intelligent system.

Visualization

MEXAR Interaction Module seems to provide a user with satisfactory representations of the main entities of the MEX-MDP domain. Problem and solution representation provides clear and detailed information that eases the user tasks. The use of payload timeline metaphor is a quick and useful mean to guarantee an immediate overall vision of salient features. On the other hand, an inadequate and sometimes missing description of the algorithm features and parameters caused a general confusion and consequently a skeptical reaction of users who preferred a random selection rather than a reasoned choice. This suggested us to design a more effective dialogue that allows a user to easily and profitably select/configure the algorithm parameters. Given that the final user of the system does not have the needed knowledge to infer the meaning of the algorithms and their parameters, the interaction module should translate this information in a comprehensible way, by using abstractions or metaphors, while hiding technical and useless details (see the *glass box* principle in Section 4.1).

A second interesting issue that is worth emphasizing is the appreciated possibility to choose between different representations of the same information. Users seemed very interested in the alternative graphic representation, as it provides a higher level and more immediate view of the problem and of the solution. The idea of organizing information in different layers has been positively accepted. After consulting the graphic representation, users used the detailed textual representation to check their answers or more precisely understand problem and solution features. This suggests the idea of structuring in layers also the graphic representation. In this way, as a first option a user would see a very high level and concise representation that can be however interactively expanded to obtain more precise and detailed information.

User participation

Users expressed a natural interest in using powerful algorithms that perform repetitive and complex tasks on their behalf. As mentioned in Section 4.1 algorithms perform better on conducting repetitive search steps that are not possible for a human user, while the user usually has more specific knowledge on a domain. The idea of allowing an incremental construction of the solution where a user can contribute with her/his expertise seems to be very promising and in line with the current literature in mixed-initiative system (see Section 4.2). This ambitious and arduous goal should stimulate systems developers in designing tools and functionalities that encourage a profitable human integration in the problem solving.

6. Related Work and Current Directions of Research

Interactive systems for solving planning, scheduling and in general complex combinatorial problems are becoming more and more pervasive in many application areas as space missions, rescue, air campaign or vehicle routing. In the last years, several systems have been proposed for interactive problem solving which can be considered in line with the MEXAR project, even though related with different domains. In this section, in order to compare the MEXAR system with different collaborative schemes we briefly review the main characteristics of some relevant interactive systems proposed in the literature and consider two general principles of interactive problem solving behind the MEXAR and the other systems.

MAPGEN (Ai-Chang et al., 2004) is an interactive system for planning and scheduling for the Mars Exploration Mission, SPIKE (Zimmerman Foor & Asson, 2002) is a tool generating plans of scientific observations for the Hubble Space Telescope. A more general purpose tool is COMIREM (Smith et al., 2003), an interactive system for continuous planning and resource management under complex temporal and spatial constraints. A more specific system is the one developed at the Mitsubishi Electric Research Laboratory [6] (Anderson et al., 2000), which proposes an effective and interactive schema called *human-guided simple search* devoted to the solution of a well-known and difficult combinatorial and optimisation problem, the *capacitated vehicle routing with time windows* (CVRTW) problem.

Broadly speaking all of the above systems follow two general principles for enabling collaborative problem solving schemes between system and user. First, they make solution models and decisions user-understandable, that is, they communicate elements of their internal models and solutions in user-comprehensible terms. This is what we call *glass box* principle. Second, they allow different levels of *user participation*, that is, a solving process can range from a monolithic run of a single algorithm to a fine-grained decomposition in a set of incremental steps. In the MEXAR domain, an example of such step might be the movement of a dumping operation from one window to another one, or the invocation of an explanation procedure on an oversubscribed channel window. These capabilities make a user really involved in the solving process, because she/he can switch from *micro* (local modifications, explanations) to *macro* actions (e.g., the application of a global optimisation algorithm) on the solution and always have a feedback through the user interface.

The so-called *glass box* principle applies to the MEXAR system as well as to the other interactive systems. It is a fundamental issue to allow a user to easily have a representation of the internal system models and decisions, and more important, to have feedback from her/his actions. Generally, the interface is differentiated on the basis of the kind of actions allowed to the user, in other words, it depends on the level of *user participation* within the interactive planning process. Systems as MAPGEN (Ai-Chang et al., 2004) and COMIREM (Smith et al., 2003) promote a problem solving style more centred on the idea of a system as an *intelligent black-board,* where a user can posts her/his decisions and see immediately the effects. In this context, conflict analysis and *explanation* services are fundamental "tools" for collaborative problem solving. Also *what-if* analysis capabilities are useful tools for guiding the search

6 http://www.merl.com

process and compare different partial solutions. On the other hand, even if the previous two systems allow also the use of solving strategies for obtaining a complete solution, systems like SPIKE (Zimmerman Foor & Asson, 2002) or the one developed at MERL lab (Anderson et al., 2000), promote a collaborative problem solving style more centered on the idea of *user guided search.* That is, a user can tune/focalize algorithms towards a given set of subgoals, or composes algorithms in order to make or improve a solution. Obviously, each time an action is performed, the user can have feedback through the interface. The MEXAR system allows a collaborative schema closer to the idea of *user guided search*, in fact, features like the *solution repository* and the graphical comparison for solutions, combined with a *portfolio* of algorithms for finding or improving a solution, promotes this style of interaction.

A future direction of work in the MEXAR project will be to explore methods for increasing the level of user participation. In fact, in order to resolve resource conflicts, currently it is possible only to remove some payload operations from the problem definition. However, this functionality can be enriched with *explanation services* on oversubscribed channel's windows and with the possibility of making available to users an extension of the functions developed for the local search algorithms, such that they will allow to perform local modifications to a currents solution, as the movement of a dumping activity from one window to another.

7. Conclusions

In this paper we described our experience in designing and developing an interaction environment for MEXAR, a decision support system devoted to solving a complex problem in the context of the space mission MARS EXPRESS.

At the end of this unique experience we can say that providing a useful and effective user interaction module, which allows an interactive problem solving shared between the user and the automated system, represents a problem as challenging and arduous as developing efficient automated algorithms.

Our system integrates automated techniques with complex interactive functionalities in order to address the visualization problem to the user, provide her/him with the possibility to personalize the interaction and maintain the responsibility in deciding the final solution.

In the paper we have also described the results of a usability test that helped us to discover limitation in the functionalities but also confirmed the robustness of the direction taken and enabled a number of directions for future work.

8. Acknowledgements

This work describes results obtained in the framework of a research study conducted for the European Space Agency (ESA-ESOC) under contract No.14709/00/D/IM. Our work would not have been possible without the support at ESA-ESOC of both the project officer Fabienne Delhaise and the MARS EXPRESS mission planners Michel Denis, Pattam Jayaraman, Alan Moorhouse and Erhard Rabenau.

9. References

Ai-Chang, M., Bresina, J., Charest, L., Chase, A., Hsu, J. C., Jonsson A., Kanefsky, B., Morris, P., Rajan, K., Yglesias, J., Chafin, B. G., Dias, W. C., & Maldague, P. F. (2004). MAPGEN: Mixed-Initiative Planning and Scheduling for the Mars Exploration Rover Mission. *IEEE Intelligent Systems*, 19 (1), 8–12.

Anderson, D., Anderson, E., Lesh, N., Marks, J., Mirtich, B., Ratajczack, D., & Ryall, K. (2000). Human-guided simple search. *Proceedings of the National Conference on Artificial Intelligence (AAAI 2000)*, Austin, Texas (pp. 209–216). AAAI Press.

Burstein, M. H., & McDermott, D. V. (1996). Issues in the Development of Human-Computer Mixed-Initiative Planning. In B. Gorayska, & J. Mey (Eds.), *Cognitive Technology* (pp. 285– 303). Elsevier.

Cesta, A., Oddi, A., Cortellessa, G., & Policella, N. (2002). *Automating the Generation of Spacecraft Downlink Operations in* Mars Express: *Analysis, Algorithms and an Interactive Solution Aid* (Technical Report MEXAR-TR-02-10 (Project Final Report)). Rome, Italy: ISTC-CNR [PST], Italian National Research Council.

Cohen, R., Allaby, C., Cumbaa, C., Fitzgerald, M., Ho, K., Hui, B., Latulipe, C., Lu, F., Moussa, N., Pooley, D., Qian, A., & Siddiqi, S. (1999). What is initiative? In S. Haller, S. McRoy, & A. Kobsa (Eds.). *Computational models of mixed-initiative interaction,* pp. 171-212. Kluwer Academic Publishers.

Glover, F., & Laguna, M. (1997). *Tabu search*. Boston: Kluwer Academic Publishers.

Nielsen, J. (1993). *Usability engineering*. San Diego, CA: Academic Press.

Oddi, A., Policella, N., Cesta, A., & Cortellessa, G. (2003). Generating High Quality Schedules for a Spacecraft Memory Downlink Problem. In F. Rossi (Ed.), *Principles and Practice of Constraint Programming - CP 2003, 9th International Conference, CP 2003, Kinsale, Ireland, September 29 - October 3, 2003,* *Proceedings*, Vol. 2833 of *Lecture Notes in Computer Science* (pp. 570–584). Springer.

Smith, S. F., Hildum, D. W., & Crimm, D. L. (2003). Interactive Resource Management in the COMIREM Planner. *Proceedings of IJCAI-03 Workshop on Mixed-Initiative Intelligent Systems*, Acapulco, Mexico (pp. 100–106).

Tsang, E. P. K. (1993). Foundation of Constraint Satisfaction. London and San Diego, CA: Academic Press.

Zimmerman, F., & Asson, D. J. (2002). Spike: A Dynamic Interactive Component in a Human-Computer Long-Range Planning System. *Third International NASA Workshop on Planning& Scheduling for* Space, Houston, Texas.