GMD Report 74

GMD –
Forschungszentrum
Informationstechnik
GmbH

Alfred Kobsa
Constantine Stephanidis (Eds.)

User Interfaces for All

Proceedings of the 5th ERCIM Workshop

Dagstuhl, Germany

November 28th - December 1st 1999

Special Theme:

User-Tailored Information Environments

November 1999

GMD –Forschungszentrum Informationstechnik GmbH
Schloß Birlinghoven
D-53754 Sankt Augustin
Germany
Telefon +49 -2241 -14 -0
http://www.gmd.de

Anschriften der Herausgeber/Addresses of the editors:

Prof. Dr. Alfred Kobsa
Institut für Angewandte Informationstechnik
GMD – Forschungszentrum Informationstechnik GmbH
D-53754 Sankt Augustin
E-mail: alfred.kobsa@gmd.de

Dr. Constantine Stephanidis
FORTH - AT&HCI Lab
GR-71110 Heraklion
E-mail: cs@ics.forth.gr

Die Deutsche Bibliothek - CIP-Einheitsaufnahme
User interfaces for all : proceedings of the 5th ERCIM workshop, Dagstuhl, Germany, November 28th - December 1st 1999 ; special theme: user tailored information environments / GMD – Forschungszentrum Informationstechnik GmbH. Alfred Kobsa ; Constantine Stephanidis (eds.). - Sankt Augustin : GMD – Forschungszentrum Informationstechnik, 1999
(GMD Report ; 74)
ISBN 3-88457-969-X

ISSN 1435-2702
ISBN 3-88457-969-X

ERCIM WORKING GROUP "UI4ALL"

5th ERCIM WORKSHOP
ON
"USER INTERFACES FOR ALL"

Dagstuhl , Germany, Nov. 28th - Dec. 1st 1999

Special Theme: "User-Tailored Information Environments"

Workshop Chair
Dr. Constantine Stephanidis

Programme Chair
Prof. Dr. Alfred Kobsa

Local Arrangements
Dr. Michael Pieper

Programme Committee
Dr. Constantine Stephanidis, ICS-FORTH, Greece (Workshop Chair)
Prof. Alfred Kobsa, GMD, Germany (Programme Chair)
Dr. Michael Pieper, GMD, Germany (Local Organiser)
Dr. Elizabeth Andre, DFKI, Germany
Prof. David Benyon, Napier University, United Kingdom
Prof. Noelle Carbonell, CRIN-CNRS & INRIA, France
Dr. Pier Luigi Emiliani, CNR-IROE, Italy
Dr. Giorgio Faconti, CNR-CNUCE, Italy
Mr. Seppo Haataja, NOKIA Mobile Phones, Finland
Dr. Ilias Iakovidis, European Commission, EU
Dr. Dominique Scapin, INRIA, France
Prof. Cristian Stary, University of Linz, Austria
Prof. Jean Vanderdonckt, Université Catholique de Louvain, Belgium
Dr. Annika Waern, SICS, Sweden
Dr. Michael Wilson, RAL, United Kingdom
Dr. Juergen Ziegler, FhG-IAO, Germany

Foreword

The 5th ERCIM Workshop on "User Interfaces for All" builds upon the results of the previous four Workshops held at ICS-FORTH, Crete, Greece, 1995; Prague, Czech Republic, 1996; Obernai, France, 1997; and Stockholm, Sweden, 1998. The electronic proceedings of all workshops, including this one, are available at http://www.ics.forth.gr/ercim-wg-ui4all .

The visionary goal of *User Interfaces for All* is the realization of the *Universal Design* principle in the field of Human-Computer Interaction. A centerpiece of this endeavor is the development of user interfaces to interactive applications and telematics services that provide access and quality in use to potentially all users. The targeted user population includes people with different cultural, educational, training and employment backgrounds, both novice and experienced computer users, the very young and the elderly, as well as people with different types of disabilities and people in various interaction contexts and scenarios of use.

At the core of the vision of *User Interfaces for All* lies a mix of existing and emerging technologies that are likely to shape future user interfaces to a wide range of applications and telematics services. These interfaces must be both *accessible* and *usable,* i.e., provide high quality interaction to potentially all users, so that they are *acceptable* by the broadest possible end-user population. This year's Workshop encouraged the submission of research reports on *User-Tailored Information Environments* that allow all users to access information that is catered to their needs with as little effort as possible.

The Workshop attracted very strong interest within, but also beyond, the European Community. Following a stringent peer review process, 13 long papers, 6 short papers, and 6 posters have been accepted for presentation at the Workshop and inclusion in the present Proceedings.

We would like to thank all contributors and participants who have made this workshop a successful international event. In particular, we wish to thank the Scientific Program Committee, Local Chair Michael Pieper, and the two invited speakers Jon Gunderson and Hans-Heinrich Bothe for their contributions in ascertaining the scientific quality of this Workshop. Henrike Gappa, Stephanie Mermet, and Bernd Murchner helped organize and run the meeting. The European Research Consortium in Informatics and Mathematics and GMD – German National Research Center for Information Technology gave generous financial support.

Alfred Kobsa and Constantine Stephanidis

Content

s = short paper
l = long paper

The 'Technology Push' and The User Tailored Information Environment

Mary P. Zajicek and Albert G. Arnold*

School of Computing and Mathematical Sciences, Oxford Brookes University, Oxford, OX3 OBP, UK. Email: mzajicek@brookes.ac.uk
*Faculty of Technology, Policy and Management, Delft University of Technology, Delft, The Netherlands. Email: a.g.arnold@wtm.tudelft.nl

Abstract

The authors of this paper wholeheartedly support the International Scientific Forum 'Toward an Information Society for All' initiative. We aim to contribute to discussion of the 'users' trajectory towards an Information Society for all as defined in 'Toward an Information Society for All: HCI Challenges and R&D Recommendations' (Stephanidis et al, 1999). A broadening of the definition of user requirements for the user tailored information environment, is suggested, to include off-line support for those members of the community who do not have the personal capital to support participation in the Information Society. These individuals are often reluctant to become involved in Information Technology and lack confidence. They find it threatening and difficult and frequently lack the resources to see ahead to the benefits that will accrue. These issues should be addressed if we are to 'push' the use of Information Technology into these previously excluded populations. This paper discusses theoretical approaches to the problem and drawing on experience gained while working with two such user groups, proposes an initial framework of measures to support their take-up of Information Technology.

1. Introduction

This paper addresses the challenges involved when attempting a 'technology push' for groups in the community who have no points of reference with Information Technology (Arnold and Vink, 1999).

It draws upon experience gained while working with two very different non-technology enabled user groups, elderly visually impaired people and those running small family hotels in Oxfordshire, UK. Although very different, both groups were involved in making the transition from non-computer literacy into the world of Information Technology and represented individuals with varying degrees of commitment to change and motivation. Similarity emerged in terms of the support needed for them to participate in the Information Environment even though their personal resources for dealing with learning were different.

Our experience shows that we cannot assume compliance in learning in these groups in which many are reluctant to participate in the Information Environment and are unable to envisage the advantages in doing so.

The central question is how can we increase user confidence to make the transition towards the world of Information Technology. This question is related to concepts of acceptability, usefulness, or utility. Davis et al. (1989) has developed a technology acceptance model (TAM), which is based on the theory of reasoned action (TRA) describing the determinants of consciously intended behaviours (Ajzen & Fishbein, 1980). According the TAM actual system use is dependent on the behavioural intention of the users to use. This intention is created by a positive attitude towards the system which stems from a cognitive evaluation process based on beliefs and norms.

According to Davis 'perceived usefulness' and 'perceived ease of use' are strong beliefs in the attitude forming process. Perceived usefulness is defined as the prospective user's subjective probability that using a system will increase his or her job performance within an organisational context. Perceived ease of use is defined as the degree to which the prospective user expects the target system to be free of effort (Davis, 1993).

This model might be valid for interactive technology, which will be used in organisational contexts. However, for the use of information technology TAM appears to be too simple, i.e. more factors might play a role. Prospective users of information technology have more freedom to choose between various applications. They are interacting with information technology in a wide range of contexts, and often lack adequate user support. In their cognitive evaluation process various beliefs and norms will play a role. For example, an important assessment has to be made about expected task demands and their coping capabilities. Does the user feel that the job can be done using Information Technology? Of course personal user needs (e.g. the need for achievement) should also be taken into consideration. Furthermore, the characteristics of the prospective application should be evaluated with regard to the task in hand, e.g. accessibility, usability, security, and reliability should be weighed up.

One could argue that two hypotheses are being made by the user: 1) the effort and the use of the application will result in the desired performance, and 2) the desired performance will lead to the rewards expected (cf. expectancy-value theory; Vroom, 1968). The user is also influenced by his/her social environment. If they are surrounded by people who are already using information technology, they are likely to experience an additional push to become a IT user.

This paper sets out recommendations for supporting the transition of non-technology enabled people into the Information Environment. In particular, to increase the users' capabilities to the point at which they feel confidence in the use of Information Technology. Furthermore, a widening of the scope of user requirements is suggested to include the technical, emotional and physical support of new computer users.

2. Non-Technology Enabled User Groups

2.1 Case1: Those running small family hotels

One author was employed by The Southern Tourist Board, UK, in the project 'Tourism Means Better Business' (Zajicek et al, 1998), to investigate reasons for the disappointing uptake of Information Technology among people running small hotels, hereafter referred to as small operators or operators, in the Oxford area. The aim of the project was to increase the effectiveness of the hospitality industry in Oxfordshire by the introduction of Information Technology to small family run hotels. It was motivated by the belief that increased use of Information Technology, for example gathering customer information and manipulating it in order to target groups with special room rates etc., would increase room occupancy in these establishments. The number of guests staying in the Oxfordshire area would be increased and thereby generate more tourist based income for the whole area. Operators were offered PCs at a subsidised rate and encouraged to use standard office software, database, spreadsheet and word processor. They were interviewed in their hotels to determine their attitudes to Information Technology, what problems they faced in coping with it and to what extent they had adopted it.

Results of the study showed that 60% of small operators welcomed the chance for more training in the use of Information Technology, although many felt they do not have enough time to attend courses as they have to be at work to cater for drop in trade. In many cases they had actually bought a computer and attended courses on office software familiarisation. However the information they acquired by attending courses was difficult to relate to the computerisation of their own business. Those attending day courses found it difficult to move forward with the knowledge gained from the course. They were taught how to use software on computers set up in the training establishment. They became proficient in the use of the software on the machines provided, but did not develop a sufficiently strong framework of the general concepts behind computer organisation to enable them to function on their own. The user group comprised mostly individuals with very low levels of formal education. They did not possess the analytical/learning skills needed to organise their own learning of software and use of the computer. They lacked confidence in learning from their mistakes and had no conceptual framework in which to work.

The following points encapsulate the off-line support needs of small establishment operators.

- Training should be stepped at different levels with a clear indication of what will be taught and at what level.
- Training should be organised at a time to suit the operators.
- Operators welcomed home based learning wherever possible since they are effectively tied to their establishments.

- Teaching material in several media, video, on the web over the telephone and face to face would be welcome.

The following hotel based support was welcomed by operators:

- The use of videos, Web materials and home tutorials to introduce Windows and the concepts behind computers.
- Step-by-Step videos, Web materials and home tutorials for home teaching about computerisation of their business.
- Teaching material supported by home visits by PC experts to see how operators are using their computers and to suggest ways forward.
- Network of operators from similar establishments to contact from work and develop supportive relationships with.

The following centrally based support was welcomed by operators:

- Help line manned by PC experts open at set times for advice and step through instructions.
- Stepped courses with a clear definition of the skills that will be learnt, the skills required to benefit from the course and expected progression.
- Central organisation of a network of operators from similar establishments.

2.2 Case 2:Elderly visually impaired people

Mary Zajicek is leader of the Speech Project at Oxford Brookes University where a study was performed to evaluate the use of BrookesTalk (Zajicek et al, 1999), a Web Browser for the blind and visually impaired, by non-technology enabled elderly people with a serious visual impairment.
These people were self-selected in the sense that they (or their family on their behalf) requested a copy of BrookesTalk after hearing about it through the media.

Evaluation of their use of BrookesTalk was performed on-line by means of email based questionnaires and by telephone. It was found that 65% of the group were unable to get up and running with BrookesTalk. They found it difficult to conceptualise the workings of a computer application. The nearest model to computer software that they could find, in order to draw comparisons, was a VCR. Many users assumed that you just had to know which button to press and it would 'work'. The concept of having a dialogue with the computer and learning to use its language was new to many participants. Many problems encountered were also due to a lack of conceptual models of the World Wide Web (Zajicek et al, 1999) and a lack of understanding of the relationship between function keys and functions.

BrookesTalk has limited functionality compared with standard visual browsers operating as it does with twelve function keys. The Microsoft Corporation has recently funded a project to integrate BrookesTalk's web summarising capabilities with Microsoft Explorer. This will provide increased functionality such as bookmarks and email and increase the complexity of use. This will benefit young technology-enabled blind people who will gain access to the functionality of Internet Explorer through BrookesTalk. However this approach compounds the problems of the elderly who require less functionality in order to learn. As a result of the study with the elderly a simplified version of BrookesTalk is under development with stepped familiarity points which allow users to consolidate their knowledge every time a new facility is learnt.

Researchers interviewing elderly users over the telephone found that they did not have the skill or confidence to try out functions to see how they work. Impaired memory is a disadvantage in exploratory learning. Elderly users were therefore not able to employ the usual suck-it-and-see method for finding out how things work and to build up a conceptual model of a system.

3. Recommendations

The users described in Section 2. do not possess the necessary personal capabilities to enable them to benefit from the Information Technology Environment. Their learning skills in this medium are underdeveloped and they do not have a strategy for learning in a concurrent software environment. As a consequence their confidence is low. Recommendations made by the International Scientific Forum imply that the solution for non –technology enabled users lies in the provision of simple easy to use interfaces. We argue that this group of users requires a technology bridge integrating familiar learning support mechanisms such as help lines or mentors and stepped learning. The users described above were superficially speaking disadvantaged for different reasons. However from a theoretically point of view they were all lacking the necessary capabilities to cope with information technology. And therefore the remedy for getting them going was similar. Currently user support requirements for the 'technology push' is not well understood and more study is required. However the following comprise some preliminary suggestions:

1. Impose a sequential structure on the concurrency of the system being learnt, albeit temporarily, in order that the main concepts can be absorbed.

2. Provide a sequence of topics/functions to learn with manageable steps between them

3. Insist on total familiarity with one topic/function before the learner moves on to the next

4. Plan the acquisition of competencies to reinforce users' developing conceptual models

5. Learning for these people is by ‘demand’. They are not skilled at absorbing large amounts of information and organising it to fit their needs. The information should be pre-packaged, but flexible to their needs.

6. Provide off-line support, on the users’ own terms wherever practicable, at the time and place of their choice.

7. Facilitate user driven learning. Provide flexibility so that wherever possible users can follow their own learning trail.

8. Base learning on exercises related to the users’ sphere of interest to facilitate awareness of what is possible.

9. Visit the user to see how they are using the system and make suggestions for progression.

10. Use different media for presenting information so that users can choose the most appropriate for them.

4. Conclusions

We see that those who use Information Technology are usually those who have the confidence to explore the system which they are trying to use where complex functionality can be discovered by cruising menus and trying things out.

If we are to ‘push’ technology use onto individuals who are currently excluded, we must interpret user requirements capture and system design in the widest sense to include outreach and whatever support measures that are needed. In addition current technology acceptance measures do not, at first sight, appear to include sufficient contextual factors to be able to contribute to an understanding of this group. The sphere of application in which they are useful should be extended.

The challenge ahead is to come to a greater understanding of the needs of non technology enabled people, so that we can support them in joining the Information Technology Society. It is vital that those who are rapidly becoming marginalised in society through their lack of understanding of Information Technology should be included. There is a danger that by enhancing usability and inclusivity in the Information Technology Environment, we further disadvantage those that are not included. Those who are outside and reluctant to join in should be drawn in on their terms.

5. References

Arnold A and J. Vink (1999) Broad Spectrum approach and Information Society for all. In: H-J. Bullinger & J. Ziegler (Eds.) *Human-Computer Interaction. Communication, Cooperation, and Application Design, vol. 2.* Mahwah, New Jersey: Lawrence Erlbaum Associates, Publishers, 757-761.

Davis, F.D. (1993) User acceptance of information technology: system characteristics, user perceptions and behavioral impacts. *International Journal of Man-Machine Studies*, 38, 475-487.

Fishbein, M. & I. Ajzen (1975) *Beliefs, Attitudes, Intention and Behavior: An Introduction to Theory and Research*. Reading: Addison-Wesley.

Stephanidis C. et al, (1999), 'Toward an Information Society for All: HCI Challenges and R&D Recommendations', *Internation Journal of Human-Computer Interaction*, 11(1), 1-28.

Vroom, V. H. (1964) *Work and Motivation*. New York: Wiley.

Zajicek M., Wheatley B., Winstone-Partridge C., (1998), Improving the Performance of the Tourism and Hospitality Industry in the Thames Valley, Technical report no. CMS-TR-99-04, School of Computing and Mathematical Sciences, Oxford Brookes University

Zajicek M., Powell C. and Reeves C., (1999), Evaluation of a World Wide Web scanning interface for blind and visually impaired users, HCI International'99, Munich

What Users Expect from Future Terminal Devices: Empirical Results from an Expert Survey

Castulus Kolo Michael Friedewald

Fraunhofer Institute for Systems and Innovation Research
Breslauer Strasse 48, D-76139 Karlsruhe, Germany
Voice: +49 721 6809-140, Fax: +49 721 6809-131
email: {cak,fri}@isi.fhg.de

Abstract

For the acceptance of future network-based media services the technical advancement of the man-machine interface is of great importance. The Fraunhofer Institute for Systems and Innovation Research (ISI) in Karlsruhe has conducted an expert survey on different acceptance factors, on the future development of media usage behavior, on network infrastructure and terminal devices as well as on the expected market development. Drawing on the results we point out perspectives in the light of today's user needs and the technical possibilities.

Over the last two decades progress in software engineering and hardware miniaturization have turned the computer from a large device that could only be operated by specialists into a common consumer good that can be found in almost all areas of private and business life. The IBM Personal Computer (1981) and the Apple Macintosh (1984) were important early milestones on the way to the computer as a useful product for everyone. Today it is much more than a simple calculating machine or tool. With the establishment and commercialization of computer networks since the early 1990's the computer became a new medium fulfilling information and communication needs [Friedewald 1999].

According to expert opinion online media services will lead to a substantial expansion of financial and time expenditure of private households for the use of media. Network-bound media offerings like Internet or online services enable easy access to conventional media contents, for example newspaper articles, music titles, and video sequences at each time of day. The most important progress however is the possible integration of different types of media as well as their linkage to additional services. This includes communication with other users, the handling of transactions, e.g. electronic shopping, and even interactive television.

Supported by the German Federal Ministry for Education, Science, Research and Technology and in co-operation with a research association focussing on media education (Medienpädagogischer Forschungsverbund Südwest) the Fraunhofer Institute for Systems and Innovation Research (ISI) in Karlsruhe has conducted a survey on the expected future development of media usage behavior, on network technology and terminal devices, as well as on market development in Germany.

The survey is part of the research project “Development of Media Services“ (DeMeS) about the future media use in the private sector [Harnischfeger et al. 1999]. 281 experts have participated, 40 % of whom belonged to the German industrial sector (media industries, information and communication technology). In addition, experts were chosen from German Institutions, who either have a scientific interest in media or who are concerned with the shaping of the media sector and its social, political and economic conditions (politics, churches and unions, as well as professional organizations). The survey was conducted by mailing questionnaires. Most of the questions were closed, showing a variety of possible answers to be ticked. The experts were not asked to justify their estimates. However several questions focusing on related topics allowed a check for consistency. Furthermore comparison of estimates by experts from different professional backgrounds showed no significant difference concerning the results discussed below. In combination with the 1998 Delphi study questioning more than 2,000 experts on the global development of science and technology [Cuhls et al. 1998], this data allows a thorough view on what users require from future terminal equipment, and what technical developments can be expected.

1 Diffusion of Terminal Devices

While in the middle of the 1970 there were no more than 165,000 computers world-wide, this number grew up to 200 million devices in 1997 [Phister et al. 1983, Kelly 1997]. Today a personal computer is available in more than 40 % of the US-American and in approximately 30 % of the Western European households. It is expected that in Europe this number will increase by further 10 % up to the year 2001 [Zerdick et al. 1999, EITO 1999].

Until recently stationary PCs were the only devices for using online services. In the future a multiplicity of new devices will appear: e. g. Network PCs that receive their software via network connection, television sets with computer functions or enhanced mobile phones. Because of particular features, such as the small display of mobile phones, the mode of interaction has to be adjusted for a number of new terminal devices [Oliphant 1999]. Media services based on the Internet Protocol Suite offer new application perspectives even for those users, who in the past did not see a reason for buying a PC with its stand-alone applications. Altogether the experts expect that the share of households with terminal devices for the use of online services will rise from 9 % today to more than 40 % in 2015. The growth will be even faster with mobile devices, which at present constitute a share of less than 1 %. It is expected that in 2015 almost 17 % of the households will be using mobile equipment (Fig. 1) [Harnischfeger et al. 1999].

The crucial point for this development is the users acceptance. Before discussing the technical factors with respect to acceptance it is worth mentioning that the social environment and the user competence are central as well. More than 80 % of the experts regard vocational practice with multimedia technologies, the integration of multimedia into the educational context, a high-quality and reasonable priced customer service as well as personal non-commercial help in case of problems as important or very important for the acceptance. These findings illustrate that the formation of user competence by self-instruction or “trial and error” that is still often practiced today will not be sufficient for a broad acceptance and future diffusion in the view of the experts.

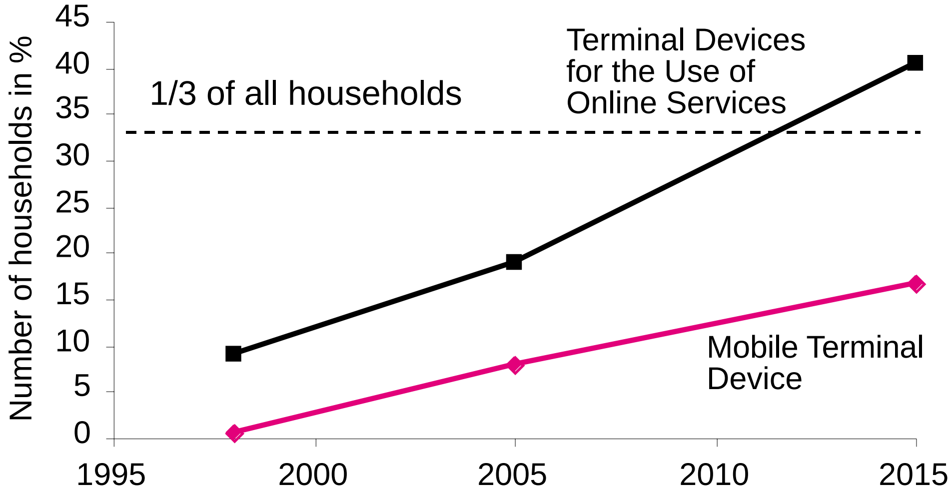


Figure 1: Expert estimate for the diffusion of terminal devices [Harnischfeger et al. 1999]

In this respect also the technical development is challenged to achieve a new quality of user friendliness.

Furthermore, the usefulness of new media services will be compared with conventional alternatives. In comparison with newspapers or television, more than 80 % of the experts named the timeliness of information, the greater flexibility in time as well as the reliability as central criteria. Attractive contents are a basic precondition, while evaluated differently depending on the user segment.

2 Technical Features and User Acceptance

Regarding the importance of hardware and software design the following picture can be drawn. Some of those aspects also concern the network infrastructure, as shown in figure 2.

Better user support is regarded as the one central requirement for the improvement of present media services. At the top of the list is the demand to reduce existing response time (99 %). With glass fiber technology and efficient transmission methods such as ATM (Asynchronous Transfer Mode) or ADSL (Asymmetric Digital Subscriber Line) this is no longer a problem. However, the connection of every household to such a high-speed network will take time, mainly due to the high supply costs for the private user.

The second most important aspect and a challenging task for software developers and designers are self-explanatory functions that reduce the time users require for training (98 %). An equally important aspect is the quick and easy software installation (98 %). This comprises sale and update of software over the Internet and the adjustment of software to individual needs.

Much importance is attributed to the application of intelligent, adaptive software (87 %) and so-called “software agents” (84 %), in order to achieve a better fit of media offerings to individual needs. The importance of large and flat displays (83 %) as well as of new generations of portable

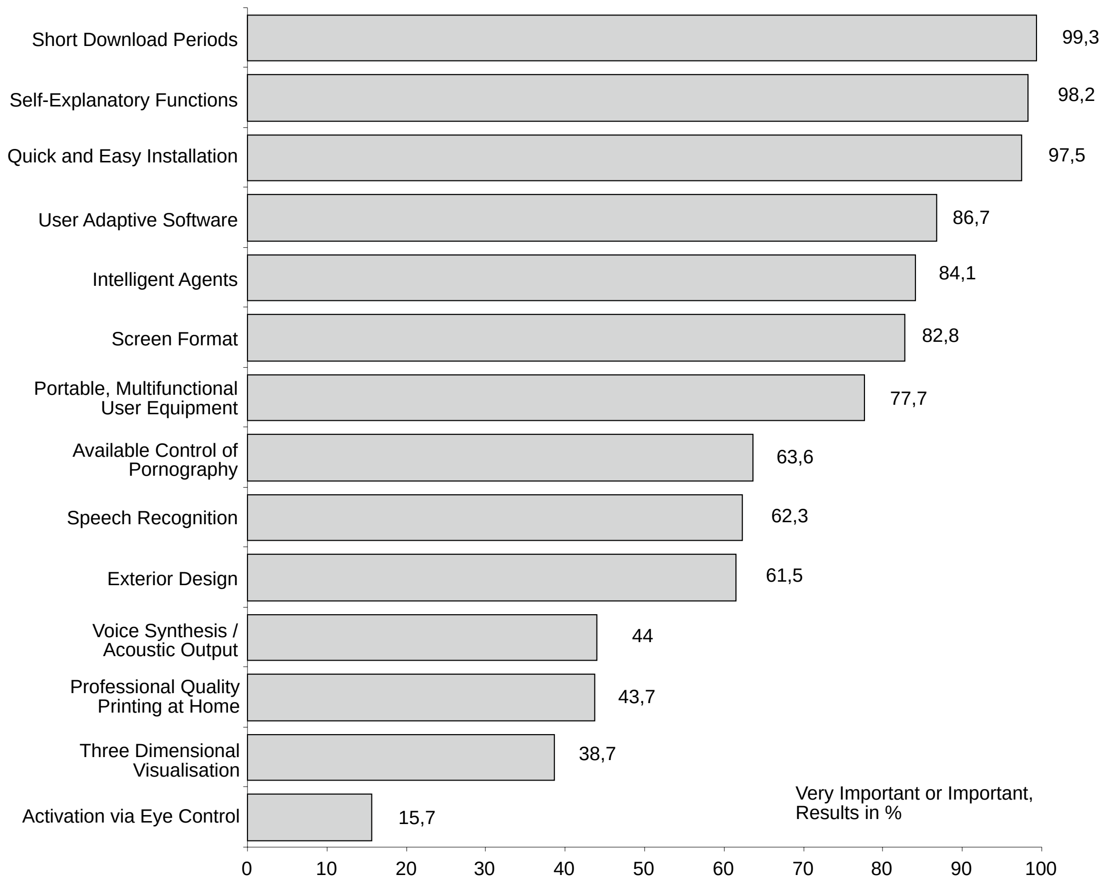


Figure 2: Importance of hardware and software design for the acceptance [Harnischfeger et al. 1999]

multi-functional devices (78 %) is also underlined by the experts. The majority of them states that acoustic output in an individually selectable language (44 %), three-dimensional visualization (39 %) or the activation and control of the PC by eye movement (16 %) are of fewer importance for the users acceptance.

It cannot be denied that the requirements for the properties of information technology will grow and become more sophisticated on the way to a knowledge society. Limitations in quantity and availability of information have been replaced by limited resources of its reception and evaluation. In addition to this quantitative aspect there is the problem of highly heterogeneous information, requiring systems that support the user in structuring information adequately. The growing requirements implies a technology that can adapt to various situations and take over tasks independently. It is not possible to meet these requirements simply by increased switching rates or further miniaturization of the computer hardware. Moreover new qualities in information processing technology is needed [Kolo et al. 1999].

3 Innovative Terminal Devices – Estimated Realization Time

For designing future terminal devices that are adaptive to individual needs, it is necessary to model the cognition of meaning and context of information at least to some extent according to human capabilities. This goal is aimed at since long by AI research with slow progress and frequent throwback. Nowadays researchers seek to complement the purely syntactical level by the semantic and the pragmatic dimension of cognitive processes, i.e. knowledge of implicit relationships and pragmatic knowledge. These aspects of human reasoning however are still poorly understood even by cognitive scientists [Winograd and Flores 1986]. This explains why experts expect breakthroughs only in the distant future (see fig. 3).[1]

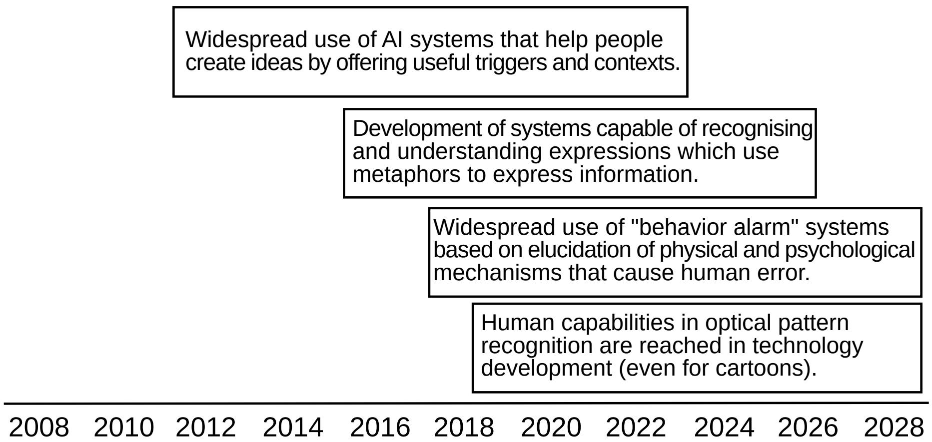


Figure 3: Expected realization timesfor intelligent applications modeling human cognitive processes [Cuhls et al. 1998]

On the other hand the expert survey underlines, that intelligent devices are of particular importance for the acceptance of online media services. Here a discrepancy between the requested features of terminal devices and reality prevails. But nevertheless terminal devices, for surfing on the net, will change on the way to the far and perhaps never reachable goal of modeling human cognitive processes. Technical innovations expected by the experts in the years ahead can be divided in three domains: I/O technology, new functionalities and the decreasing dependence on a fixed location.

3.1 Input/Output Technology

According to expert opinion I/O technology is the most advanced among these domains. Despite the long research tradition in I/O technology, so far mainly the development of two-dimensional graphic I/O was advanced. Today engineers increasingly concentrate on multi-modal man-machine interfaces enhanced through I/O procedures in natural language as well as on tackling the third dimension (fig. 4) [Myers 1998, Glinert et al. 1996].

[1] The time frames given in this and the following figures correspond to the first and the last quartile. 50 % of all expert judgements lie in this time interval.

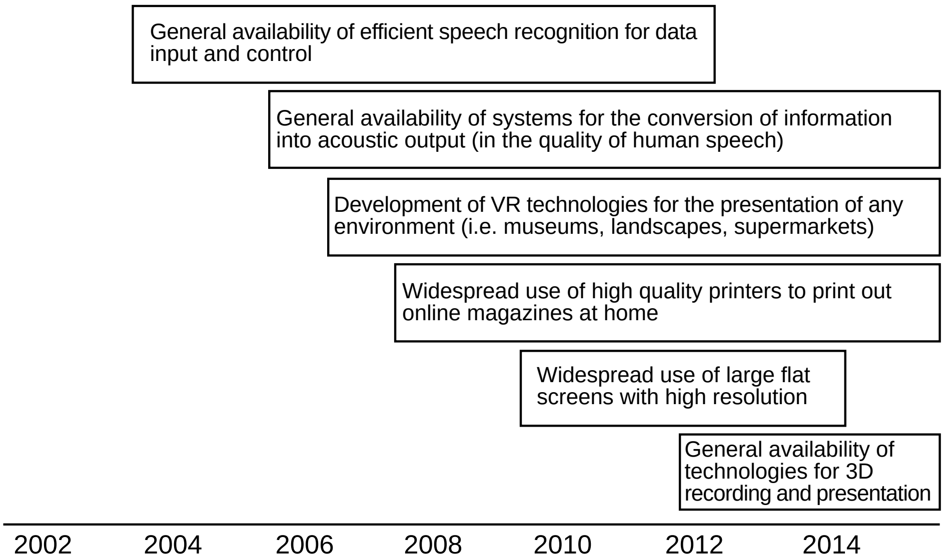


Figure 4: Expected realization time for innovative I/O technologies at the man-machine interface [Harnischfeger et al. 1999]

According to the experts powerful devices for speech recognition and speech synthesis could be widespread already in the next five to ten years. In the same period the development of VR technologies for the presentation of any environment is considered probable. General availability of technologies for 3D recording and presentation however is not expected before 2012.

3.2 New Functionalities

Even more important than I/O technologies with respect to the user acceptance are new functionalities for terminal devices that are not necessarily resident in the terminal device but could also be provided via network connection. This comprises simple installation and control as well as adaptivity to the users needs. These properties are tried at with software agents for some years now. Software agents are even considered the key technology for online services (i.e. World Wide Web) [Klusch 1999]. Software agents are a class of intelligent programs that access autonomously heterogeneous, geographically dispersed data and support the user in selecting relevant information. Such programs protocol and evaluate user activities. New or shifted interests are automatically recognized and taken into account for subsequent searches [Maes 1995]. Ideally, software agents release users from routine activities leaving more time for productive and creative work. Already 30 years ago software pioneer Douglas Engelbart had the vision that such applications could selectively augment human intellect leading to a symbiotic relationship of man and computer [Engelbart 1995].

3.3 Mobile Terminal Devices

Innovations from different fields of research will lead to an increasing diffusion of mobile terminal devices for online media services during the next 15 years (Fig. 6). The development

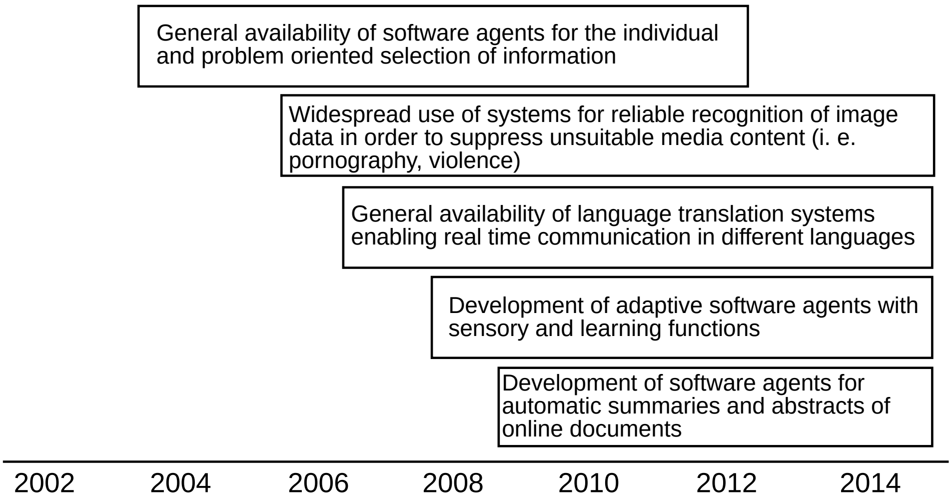


Figure 5: Expected realization time for innovative functionalities [Harnischfeger et al. 1999]

of electronic components with low power consumption and light, powerful batteries are considered important. Their use, however, will be widespread comparatively late, according to the experts. The currently developed UMTS (Universal Mobile Telecommunication-System) standard with a transmission rate of 2 Mbit/s already addresses the increasing importance of broadband wireless data communication and even higher data rates are already technically feasable [Johnston 1998, IEEE Spectrum 1999]. But mobile terminal devices with rates higher than 10 Mbit/s will not be widespread among private users before 2006.

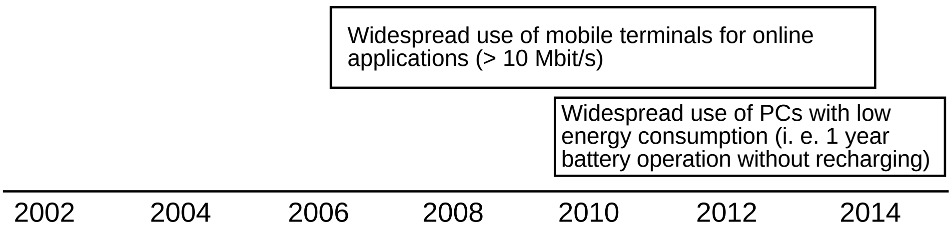


Figure 6: Expected realization time for innovations leading to enhanced mobile terminal devices [Harnischfeger et al. 1999]

4 Conclusion

In spite of the tremendous growth in the use of online services, the necessary terminal devices will change less dramatically in the coming 15 years as forecasted by some technology evangelists. The most requested features of innovative technology, in particular user adaptive devices and software agents to selectively augment human intellect will only partly be developed. Instead of new revolutionary qualities, a number of incremental improvements in transmission rates, computing power and software functions are ahead. Although these innovations will make the use of online media services more comfortable or simply more exciting, users will still have to accept the numerous minor and major annoyances whilst working with their terminal devices. The still increasing number of online users, however, shows that many are willing to accept these circumstances. Hence, the increase is less due to more user friendly devices but rather to new or improved aspects of utility compared to traditional alternatives. Such aspects are in particular the greater flexibility in time and the timeliness of information content. A crucial factor for the broad diffusion of online media services remains the strengthening of user competencies through education and training as well as personal, non-commercial help in case of problems.

References

[Cuhls et al. 1998] Cuhls, Kerstin, K. Blind, H. Grupp, H. Bradke, C. Dreher, D.-M. Harmsen, H. Hiessl, B. Hüsing, G. Jäckel, U. Schmoch und P. Zoche. *Delphi '98 Umfrage. Studie zur globalen Entwicklung von Wissenschaft und Technik*. Fraunhofer-Institut für Systemtechnik und Innovationsforschung, Karlsruhe, 1998.

[EITO 1999] *European Information Technology Observatory 99*. Eurobit, Frankfurt am Main, 1999.

[Engelbart 1995] Engelbart, Douglas C. *Toward Augmenting the Human Intellect and Boosting our Collective IQ*. Communications of the ACM, 38(8):30–33, 1995.

[Friedewald 1999] Friedewald, Michael. *Der Computer als Werkzeug und Medium: Die geistigen und technischen Wurzeln des Personal Computers*, Bd. 3 d. Reihe *Aachener Beiträge zur Wissenschafts- und Technikgeschichte des 20. Jahrhunderts*. GNT-Verlag, Berlin und Diepholz, 1999.

[Glinert et al. 1996] Glinert, Ephraim P., R. L. Todd, and G. B. Wise. *The Adaptive Multi-interface Multi-modal Environment*. ACM Computing Surveys, 28(4es), 1996.

[Harnischfeger et al. 1999] Harnischfeger, Monika, C. Kolo, and P. Zoche. *Future Media Use in the Private Sector: Expert interview study of the Fraunhofer Institute for Systems and Innovation Research in co-operation with and sponsored by Medienpädagogischer Forschungsverbund Südwest*. Fraunhofer Institute for Systems und Innovation Research, Karlsruhe, 1999.

[IEEE Spectrum 1999] *Special Issue on Technology 1999: Analysis and Forecast*. IEEE Spectrum, 36(1), 1999.

[Johnston 1998] Johnston, William. *Europe's future mobile telephony system*. IEEE Spectrum, 35(10):49–53, 1999.

[Kelly 1997] Kelly, Kevin. *New Rules for the New Economy*. Wired, 5(9), 1997.

[Klusch 1999] Klusch, Matthias. *Intelligent Information Agents, Agent-Based Information Discovery and Management on the Internet*. Springer, Berlin, Heidelberg u.a., 1999.

[Kolo et al. 1999] Kolo, Castulus, T. Christaller und E. Pöppel. *Bioinformation: Problemlösung für die Wissensgesellschaft*. Physica-Verlag, Heidelberg, 1999.

[Maes 1995] Maes, Pattie. *Intelligent Software*. Scientific American, 273(3):84–86, 1995.

[Myers 1998] Myers, Brad A. *A Brief History of Human-Computer Interaction Technology*. Interactions, 5(2):44–54, 1998.

[Oliphant 1999] Oliphant, Malcolm W. *The Mobile Phone Meets the Internet*. IEEE Spectrum, 36(8):20–28, 1999.

[Phister et al. 1983] Phister, Jr., Montgomery, A. R. H. Lloyd, and E. K. Yasaki. *Computer Industry*. In Ralston, Anthony and E. D. Reilly, Jr., eds.: *Encyclopedia of Computer Science and Engineering*, pp. 333–355. Van Nostrand Reinhold, New York, Second ed., 1999.

[Winograd and Flores 1986] Winograd, Terry and F. Flores. *Understanding computers and cognition: a new foundation for design*. Ablex, Norwood, 1986.

[Zerdick et al. 1999] Zerdick, Axel, A. Picot, K. Schrape, A. Artopé, K. Goldhammer, U. Lange, E. Vierkant, E. López-Escobar und R. Silverstone. *Die Internet-Ökonomie: Strategien für die digitale Wirtschaft*. Springer, Berlin und Heidelberg, 2., korrigierte Aufl., 1999.

SELECT: Social and Collaborative Filtering of Web Documents and News

The SELECT Project Team[1]

Abstract

We describe the goals and progress to date of SELECT, a project funded under the EC Telematics Applications Programme. The objective of the SELECT project is to help Internet users to find the most reliable, valuable, important and interesting information quickly and easily and reduce information overload. SELECT is aimed not only at users who search for specific information, but also at those who use the Internet to keep up to date with what is happening in particular areas. In these ways, SELECT will make a positive contribution to the problem of helping users to tailor their information environments to meet their individual needs.

The approach adopted in SELECT is based upon information filtering. SELECT makes use of two filtering techniques. One is to make recommendations derived from an individual user's past choices. The other is to make recommendations derived from the behaviour of other users through social, collaborative filtering, especially those who have displayed similar tastes and interests in the past. Both techniques make use of users' ratings of Internet documents, either given explicitly or derived implicitly from evidence of users' behaviour.

Introduction

The SELECT project is funded by the EC Telematics Applications Programme and involves partners from 9 countries. Its objective is to help Internet users to find the most reliable, valuable, important and interesting information quickly and easily, to avoid trash and reduce information overload. SELECT is aimed not only at users who search for specific information, but also at those who use the Internet to keep up to date with what is happening in particular areas. Project partners include one of the most successful European providers of Internet search services, and organisations representing more specialist user groups.

---

[1]Roland Alton-Schiedl, TUV, Vienna, Austria.
Jesper Ekhall, Euroseek, Sweden.
Olivier van Geloven, Medibridge, Belgium.
Laszlo Kovacs and Andras Micsik, SZTAKI, Budapest, Hungary.
Christopher Lueg, IFI, Zurich, Switzerland.
Richard Messnarz, Independent Software Consulting Network, Ireland.
David Nichols, Computing Department, Lancaster University, UK.
Jacob Palme and Torgny Tholerus, KTH Technical University, Stockholm, Sweden.
Rob Procter and Dave Mason, Division of Informatics, University of Edinburgh, Scotland.
Enrico Stupazzini and Massimo Vassali, Omega Generation, Italy.
Richard Wheeler, StarLab, Brussels, Belgium.
Corresponding author: Rob Procter, Division of Informatics, University of Edinburgh, Kings Buildings, Edinburgh, EH9 3JZ, Scotland. Email: rnp@dcs.ed.ac.uk

The Internet has opened up important new opportunities for knowledge exchange between scientific, technical, professional and other users. Sometimes, a user's need is to find particular information on particular topics, in other cases it is to update knowledge, to keep up-to-date with recent developments and increase contacts with other people with the same interests and specialities. There are two particular reasons why this is not easy to do. First, since anyone is allowed to put up any information they like on the Internet, there is no quality control (such as is done by editors and reviewers of magazines and journals). Second, there is often too much information, making it difficult to find what is most interesting and relevant.

It is clear that Internet users need tools so that they can find the information most valuable to them within the limited time they may have available. We believe that the free nature of the Internet is very important, however, so it is not our intention to implement techniques for censoring. Rather, our goal is to develop and implement techniques to aid users to find the information that is of the highest quality and relevance for their particular interests. In this way, SELECT will make a positive contribution to the problem of helping users to tailor their information environments to meet their individual needs.

SELECT is tackling this problem by using two main techniques. One is to make recommendations that are derived from an individual user's past choices. The other is to make recommendations derived from the behaviour of other users through social, collaborative filtering, especially those who have displayed similar tastes and interests in the past. Both techniques make use of users' ratings of Internet documents, either given explicitly or derived implicitly from evidence of users' behaviour.

The focus of SELECT is the WWW and Usenet News, the two most heavily used information domains on the Internet. The impact of the former is well known: it is estimated that several hundred million hours are collectively spent surfing the WWW per month. Usenet News is used by hundreds of thousands of people every day and generates an enormous amount of information. According to volume data published by the Swiss Academic and Research Network SWITCH, Usenet offers far more than 50,0000 articles per day and the amount is increasing dramatically.

Information Filtering

There has been considerable interest within the area of information filtering in recent times and several systems are now in use (see van Bommel, Koster and van der Weide (1997) for a good overview of ongoing research). For example, Sepia Technologies has developed a collaborative filtering system for movies, music and books.[2] Surflogic has developed Surfbot, a web browser plug-in that will search for and filter information on the net according to a user's needs.[3] The best known application of social filtering is Firefly, a commercial company that

2 http://www.sepia.com/suggestion_e.html

3 http://www.surflogic.com

keeps a database of ratings of movies and music.[4] A user can connect, input his or her favourite movie or music, and be told which other movies and music were rated highly by people with similar tastes as the user. The MIT Centre for Coordination Science has developed GroupLens, a social filtering system for Usenet News (Resnick et al. 1994a, 1994b). Amazon, the Internet-based book retailer, offers its customers the opportunity to rate books and these are then offered to potential purchasers as a guide.[5]

We believe there are several reasons why the SELECT project will succeed better than previous projects. First, our intention is not to develop just another filtering system, but a filtering architecture into which different filtering methods can be plugged, compared and evaluated against one another. Most other projects in this area have just developed one single filtering method, and they have usually not attempted to perform any large-scale evaluation of user satisfaction with it. In SELECT, users will be able to test and evaluate different filtering methods against each other and indicate their opinions of them. Second, the SELECT project involves the creation of a ratings database for use in collaborative filtering. There have been some experiments before with collaborative filtering, but most have not been able to create the database needed. In SELECT, we have enlisted a major European Internet search service provider and their customers, which will enable the project to create a ratings database on a scale not previously achieved.

An Overview of SELECT

The main focus of the SELECT project is the development of rating and filtering tools. By rating tools we mean mechanisms for users to evaluate and store their ratings of Internet documents and resources. By filtering tools we mean mechanisms to automatically scan Internet documents and resources before delivery to users. The result of filtering can be an ordering of documents and resources with the most interesting first, marking up of documents and resources with codes to help the user in manual decisions on what to read or even the discarding of less interesting documents.

Which of these approaches is used depends on users' needs. For areas important to a particular user, the filter will sort items, not discard them, but for less important areas, the user may prefer that the filter automatically discards documents. Filtering can also be based on ratings provided by the author of a document or by other readers of that document. In particular, the project is developing tools for social filtering, i.e., the selection of documents based on ratings made by people with similar values, competence and interests as the person for whom the filtering is done. This recommender group may be defined automatically by the system or users may nominate their own peer group. Users may belong to different recommender groups, reflecting their various distinct interests.

Part of the project is to develop and test different filtering methods: methods of assigning ratings to documents, methods of using these ratings to filter documents and methods of finding

[4] http://www.ffly.com
[5] http://www.amazon.com

suitable filtering rules for each user. Automatic methods, where the system derives the filtering conditions from user actions or evaluations of documents and manual methods, where interaction between the user and the filter is used to establish the filtering conditions, are also being developed. At its simplest, however, SELECT simply allows users to perform the filtering themselves, informed by the ratings that others have given documents.

Rating

Rating is implicit in most non Internet-based, traditional publishing services. For example:

- Newspapers, magazines, books, which are rated by their editors or publishers, selecting information that they think their readers will want.
- Consumer organisations and trade magazines which evaluate and rate products.
- Published reviews of books, music, theatre, films, etc.
- Peer review method of selecting submissions to scientific journals.

SELECT enables Internet users to input their evaluations of document quality. These are stored and used to aid the rater and/or other users to choose which documents to read. To improve the value of rater recommendations, the interest and knowledge profiles of users can be matched to determine which users have with the same interests, values or knowledge. For example:

1. A user with a particular religion or political affiliation may prefer to find information which has been highly rated by other people with the same personal values.
2. A specialist in an area may want to find high quality information. Information that is of high quality for the specialist may be too complex for a beginner. Information that the specialist finds trivial may be valuable for a non-specialist who wants to learn the basics about a particular topic.

Using the tools provided by SELECT, rating may be applied to many kinds of documents, such as WWW pages, Usenet News postings, email, electronic journal papers, etc. Its purpose may be to increase the quality of the documents read, or to avoid certain documents deemed unsuitable in certain communities for certain groups of readers (for example: violence, pornography).

Filtering

Filtering refers to the sifting of information according to some predefined criteria. This sifting has some similarities to information retrieval, but is also different in many aspects. In addition to the ratings given to a document by other readers (or by its author), filtering may be based on content-related criteria such as:

- Keywords in the document.
- Semantic analysis of the document.
- Analysis of the stylistic and genre qualities of the document.

- Analysis of the similarities between the document and other documents which the same user has rated highly.
- Documents directly related to other documents of high interest to the user, for example, by having hyperlinks to the document of interest.

Note that filtering is not only a matter of dividing all documents into two categories, good and bad, for a particular user. Often, what the user needs is instead a list of documents sorted by a matching index. Also, users may often want to sort information of interest into different areas representing their various interests.

Earlier studies of user requirements for rating and filtering tools have shown that different users have different requirements (Lantz 1993, Lantz 1995, Fahraeus 1997, Schmutzer et al. 1997, Irmay 1997). One user is interested in medieval religious beliefs; another is interested in particle physics. Another wants an overview of the knowledge on a certain topic; another wants to find the latest news. One user may want to get the maximum amount of information of value in a limited time, another user wants to browse and entertain at leisure. One user is an expert, another is a novice, in the subject area in which they are retrieving information.

The question is how can tools be designed to cater for all these differing users with their differing needs. However, even though users are different, they are common in that each user wants to find information of value to him or her. This is the basic user need that the SELECT project addresses. Its goal is not to find good information, according to some particular criterion of goodness, rather its aim is to develop tools that will make it easier for each Internet user to find the information that is important and interesting to them.

Rating Sources

There are many different kinds of rating with different user requirements. In some domains, people are employed for making ratings. This is very common outside the Internet; most newspapers and journals have some rating system to decide what to publish and what to omit, even if they do not use this term for what they are doing. A special case is the peer review system used for choosing contributions to scholarly scientific and technical journals and conferences. In the electronic publishing area, this kind of rating is applied by portal services, perhaps the best known of which is Yahoo.[6] In Usenet News, moderated groups publish only contributions that have been approved by one or more moderators. A big disadvantage with human moderators is the delay they cause in publishing. In newsgroups and mailing lists, the time interval between one message and a reply to it is often only a few hours; in moderated lists, this time is lengthened to usually about a week. It is obvious that this can severely hinder rapid interaction in discussions.

[6] http://www.yahoo.com

		Right to rate a document	
		Everyone can input any rating (except some limitations on rating your own documents. This rule is probably not suitable to enforce automatically.	The right to input ratings is limited in some other way, to select people most proficient at providing good ratings in some way. Selection of such people may be a problem.
Use of ratings in filtering	Some kind of statistic (average, median, upper quartile) of all ratings set by everyone or by members of your peer group i.e., members of a professional organisation or expert in a particular area.	Advantage: Lots of ratings available. Disadvantage: Ratings may not agree with your personal preferences.	Advantage: Better rating, may avoid misuse. Disadvantage: May reduce the amount of ratings available.
	Ratings of people with similar views to yourself are preferably used through an automatic mechanism of comparing your ratings with those of other people.	Complex to implement but might provide very good ratings for your views and requirements. Encourages ratings, since only by giving ratings can your preferences be matched with others. Avoids problems of designating people with good competence to provide ratings.	Combines two different ways of trying to achieve the same thing: ratings set by those providing good ratings are given priority. This combination should not be used unless carefully analysed, since otherwise the two services can interact in unsuitable ways.

Table 1: Options for sources and uses of ratings.

Some systems follow an open rating principle, i.e., they allow anyone (or almost anyone) to rate any document. Sometimes just an average of these ratings is used, but some systems (e.g., Firefly) rate documents based on other people who have similar tastes (views, values, competence). A variant of this is to put people into different categories, so that users might specify that they prefer documents rated highly by other people in their own category (political or religious group, scientist, etc.). Document authors can also provide ratings, with the advantage that more documents get rated, and that the ratings are easily transmitted with the document.

Table 1 summarises the interaction of two key dimensions of rating system design. The horizontal axis represent the options with respect to ratings submission, the vertical axis represents the options of whose ratings to use to guide the choice of documents.

After an initial investigation, it was decided that SELECT would focus on an open rating system. This raises some important issues. First, if anyone is allowed to submit ratings, there is a risk of misuse by people putting in high ratings on their own documents, or collusion between two people putting high ratings on their own documents. A check for the domain of the rater and the document can stop ratings by people in the same domain, but this is not a full protection.

People known to misuse the rating system in this way can be identified and put on a stop list. Social codes that such misuse is not permitted may also help.

The second issue is how to get people to provide ratings. A good solution to this problem is that used by, e.g., Firefly, where users have to provide ratings to get access to the ratings of others. A variant of this is that a filtering system may use the ratings by a user as a tool in developing filtering conditions.

Explicit and Implicit Ratings

Much of the current work on social filtering tools for the WWW has focused on so-called explicit methods, i.e., where the rater annotates a document (e.g., Bouthors and Dedie 1999) or (more simply) inputs a rating value. One drawback of this approach is that it calls for extra effort on the part of the rater, whilst failing to provide an equally immediate benefit, a recognised problem in collaborative systems (Grudin 1988). In contrast, implicit methods require no extra effort on the part of the rater, but have the disadvantage that the rating information provided has lower value. Some tools have attempted to find some middle ground between explicit and implicit approaches (Hill, Stead, Rosensteian and Furnas 1995). The SELECT project is pursuing the possibilities for using implicit, as well as explicit ratings, exploring how implicit approaches might be improved to provide rating information and higher value and relevance (Procter and McKinlay 1997).

In order to generate implicit ratings, it is necessary for users' behaviour to be observed. There are a number of kinds of information that can be extracted as side effects of users' browsing behaviour. These include:

1. Document read time. Morita and Shinoda (1994) reported a positive correlation between the time spent reading a document and the reader's assessment of its quality. The GroupLens project was able subsequently to verify this result for Usenet News postings (, Konstan et al. 1997).
2. Documents that the user has bookmarked. However, surveys of Web users provides evidence that they typically bookmark fewer than 50% of the pages they find interesting; bookmarks tend to be evidence of strong, rather than marginal interest, so bookmarks set a relatively high threshold for recommendations (Rucker and Polanco 1997).

In addition, relevant information can be extracted from the documents themselves. For example:

1. Keywords, either as provided by the author or extracted automatically.
2. Text/image ratio, text/image hyperlink ratio and number of hyperlinks in the current document. From this information, it may be possible to derive some notion of genre and genre types into which documents may be categorised.
3. Language of the document.

In SELECT, users may also register a profile of their interests, likes and dislikes when they begin to use the service. This will be used in combination with the above data to generate an implicit rating of the document being viewed.

An Outline of SELECT System Architecture

The SELECT architecture is based upon a client-server approach. The architecture is shown in Figure 1. The client side consists of a number of modules. The principal ones are as follows:

- User interfaces for ratings input and display.
- User interface for profile and preferences setting, e.g., interests, filtering rules, etc.
- Implicit rating module. This will work behind the scenes generating ratings derived from user behaviour and documents.

The principal server side modules are as follows:

- Passive filtering, i.e., where recommendations are simply based on the submitted ratings of documents.
- Active filtering, i.e., where recommendations are based upon the ratings of documents submitted by people with interests and/or rating histories that match those of the user. These recommender groups may be defined by the system from comparisons of previous behaviour, or may be user-nominated.
- A ratings database containing individual ratings submitted by both registered and anonymous users.
- A profiles database containing information about registered users, including their interests.

Client and server communicate using specially defined protocols. The functions supported include sending and requesting ratings, registering raters and exchanging ratings between SELECT servers.

An Example Implementation

For a service like SELECT to be of greatest use, the availability of ratings derived recommendations needs to be as unrestricted as possible. That is, the system should be capable of making recommendations in whatever context a user happens upon a document. This means that the user is not required to explicitly ask for highly recommended documents (say on a particular topic), but that the user will be able to see the SELECT service's recommendations seamlessly integrated with the circumstances in which the documents become available.

As an illustration, consider the WWW page that a user is currently reading. This page may contain links to other pages or documents. The user's problem is to choose which of these links to follow. In other words, the user needs SELECT's recommendations pertaining to these (as yet unseen) documents. One solution which satisfies the requirement for seamless access

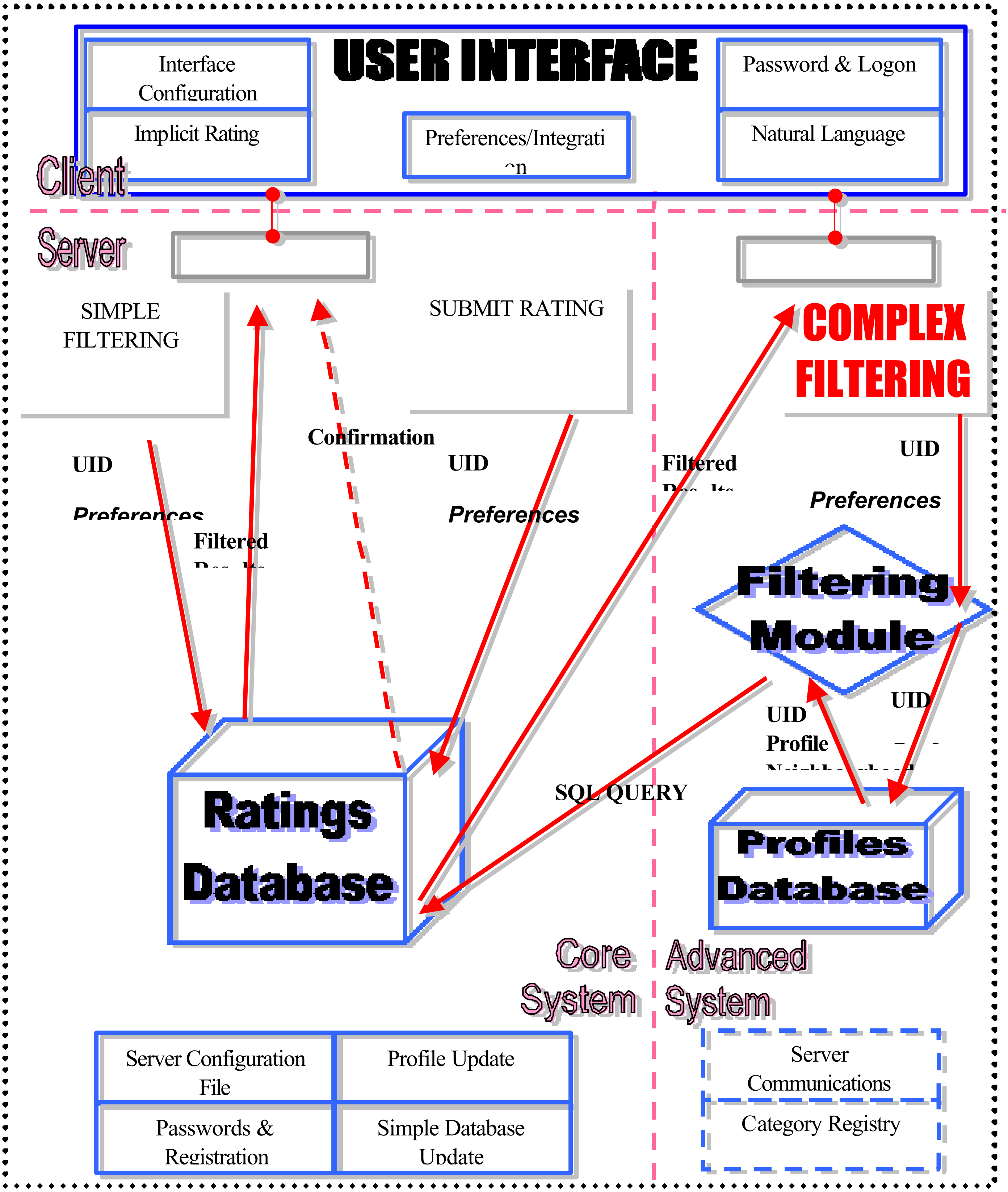


Figure 1: SELECT system architecture.

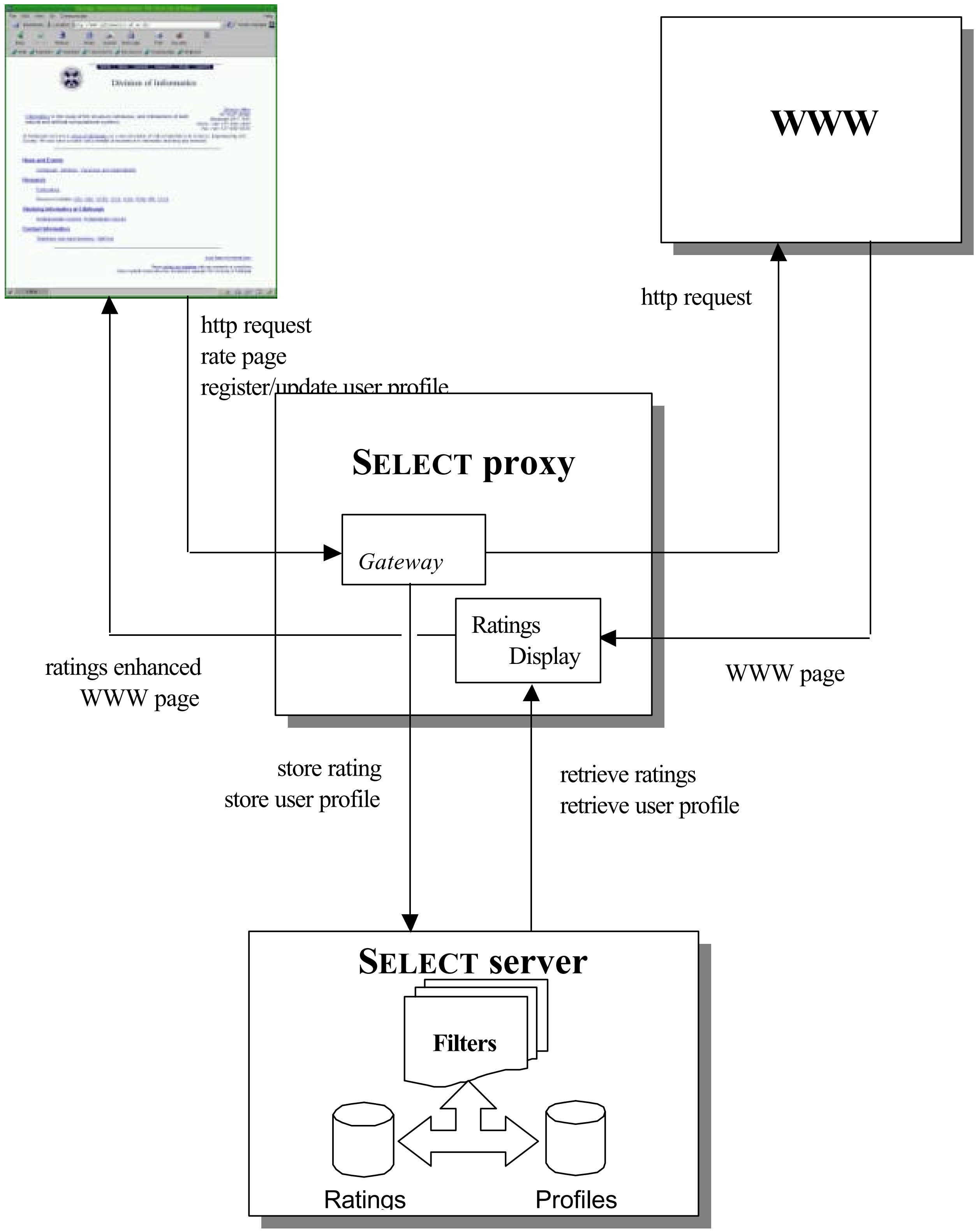


Figure 2: A proxy-based implementation of SELECT.

is to use a proxy-based approach, similar to that used, for example, in WebWatcher (Joachims, Frietag and Mitchell 1996) and Pharos (Bouthors and Dedieu 1999).

In general, a proxy-based approach allows the content of a document, say a WWW page, to be modified in a variety of ways. Content may be modified or removed before delivery to the user; there are several proxies that allow advertisements to be removed from WWW pages (e.g., Intermute[7]). Alternatively, documents can be enhanced with ratings or annotations (e.g., CritSuite[8]). In addition to single-purpose client proxies there are several existing proxies which can be extended to perform arbitrary modification of WWW content (e.g., Muffin[9]). In combination with a ratings server such a proxy can enhance WWW pages to present the user with social feedback on the quality of links in the current document.

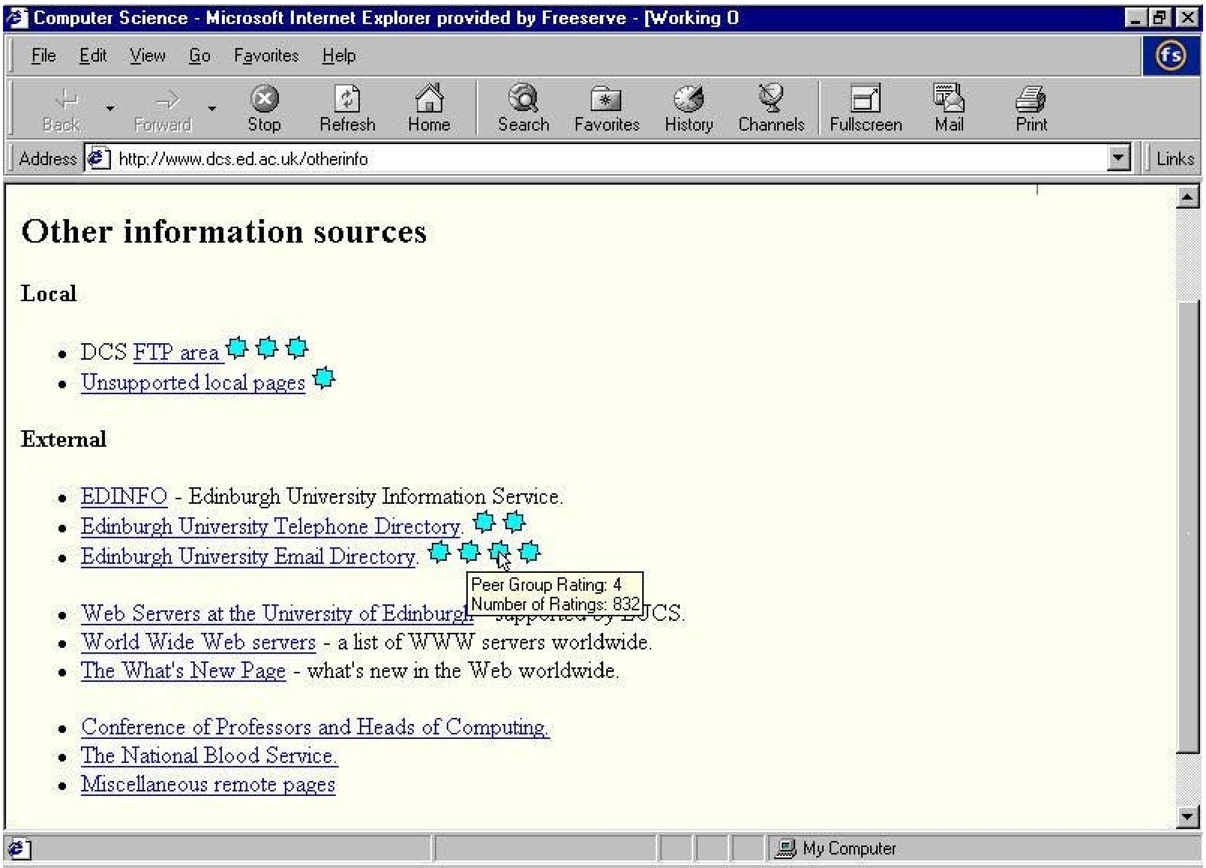


Figure 3: An example ratings-enhanced WWW page.

A proxy-based approach is sketched in Figure 2. As each WWW page request is made by the user, the proxy sends a query to the SELECT server requesting the ratings that are held for this particular document. When the WWW page is returned by its host site, relevant ratings information are returned by the proxy and are displayed on the user’s desktop. More

[7] http://www.intermute.com/

[8] Foresight Institute, http://crit.org/

[9] http://muffin.doit.org/

interesting, however, are the options that a proxy-based implementation opens up for enhancing information about the links contained within the WWW page. An example is shown in Figure 3. Here, ratings for individual links are indicated by the '*' symbols. Passing the cursor over the rating causes a floating pop-up window to appear with more details about how the rating was been derived. In the example, this indicates that it has been calculated from over 800 individual ratings submitted by members of that user's peer group.

Other services, such as site statistics, download predictions and document previewing may also be incorporated within the same basic architecture (Stanyer and Procter 1999).

User Interfaces for Ratings Input and Display

It is critical that the effort required to make an explicit rating be minimised. At the same time, it is also important that raters provide enough information for the rating to be capable of being interpreted accurately. The GroupLens project reduced user inputs to a single click by combining the rating function with the 'next article' function. Although this does not seem feasible in a WWW domain (where the user may follow any link on a page), it is indicative of the sensitivity of collaborative filtering systems to rating input costs. A lack of ratings may prove especially significant at the start of a collaborative system: where the functionality is dependent on ratings input by users then the system has few benefits to a potential user (Lueg 1998). This 'cold-start' problem is usually addressed by using content-based techniques, e.g., filterbots (Sarwar et al. 1998). In SELECT we can also consider using implicit techniques for a period before introducing explicit ratings.

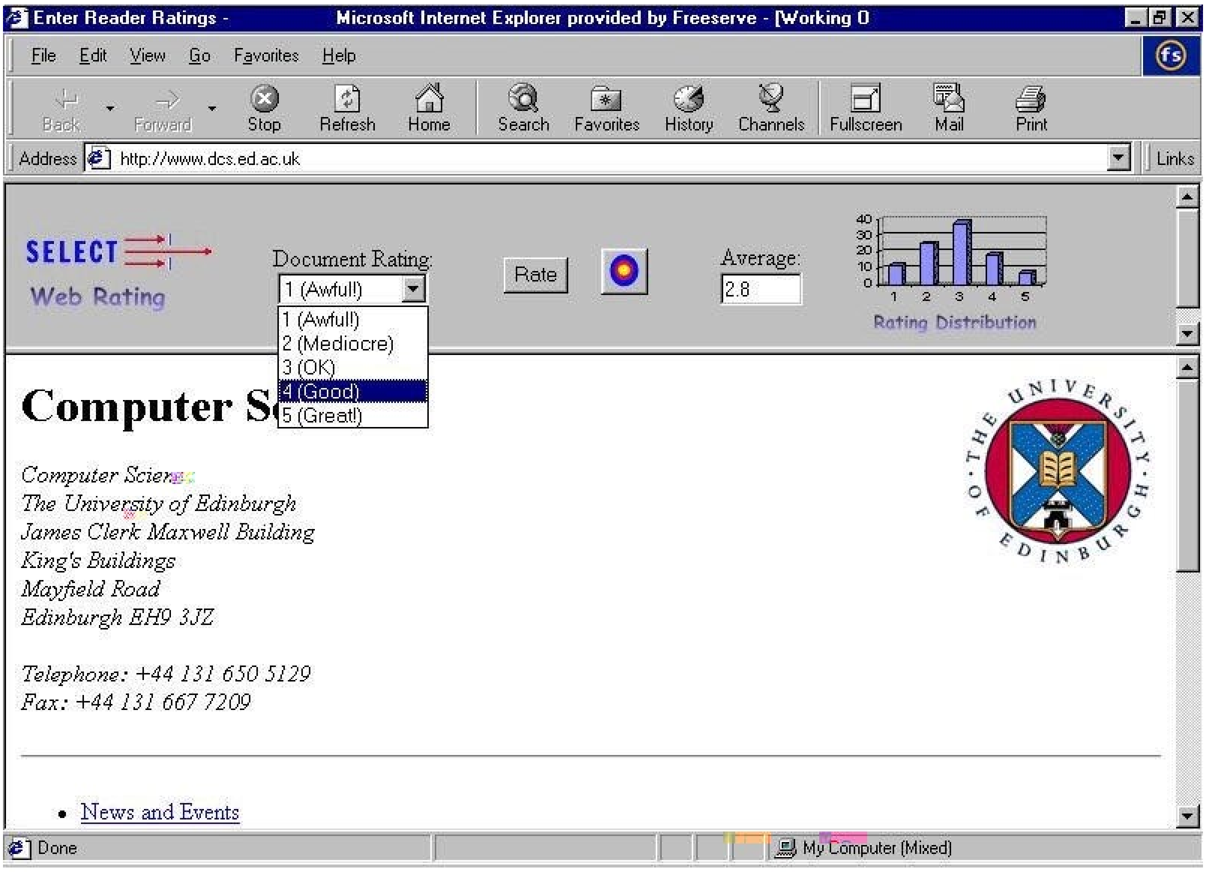


Figure 4: A prototype ratings input user interface.

Even when ratings are being provided and used to recommend resources, there may still be dis-incentives for users to provide ratings (Avery and Zeckhauser 1997). Although the user interface may enforce a 'stick' approach (not providing rated documents to those who do not contribute ratings) it can also enhance incentives for rating by using social 'carrots'. Under certain circumstances users will perform ostensibly altruistically, i.e. they may provide ratings without tangible benefits. One such intangible benefit could be 'fame' -- the user interface could show who is contributing to the community by providing ratings, or show that a particular recommendation has been derived mainly from a small group of raters.

Although it is in users' interests to provide ratings, as they help personalise any recommendations, by providing an alternative 'benefit' the SELECT user interface may be able to achieve the necessary 'critical mass' of users. The visibility of such social benefits helps balance the (possibly) 'black box' benefits of increased personalisation. In order to be part of a successful collaborative filtering system the interface then has to not only be sensitive to the cognitive costs of ratings input, but also support wider social relations. The user interface is not merely a mechanism for connecting users to the ratings database but it reflects the organisation of the virtual community and the contributions (or otherwise) of the participants. The Knowledge Pump (Glance, Arregui and Dardenne 1998) is an example of a user interface that attempts to reflect these concerns within an intranet environment; a challenge for SELECT is to generalise to the environment of the Internet.

A survey was carried out to determine which of several different ratings input user interfaces people found most acceptable. The most popular choice is shown in Figure 4. This example uses a frame-based approach, the frame being added by the proxy as above. The rating input interface incorporates a display of the WWW page's current rating, shown here as both an average and a population distribution for the individual ratings.

Summary

The goal of the SELECT project is to develop and implement techniques to aid users to find the information that is of the highest quality and relevance for their particular interests. In this way, SELECT will make a positive contribution to the problem of helping users to tailor their information environments to meet their individual needs.

The approach adopted by is SELECT based upon two main techniques. The first is to make recommendations that are derived from an individual user's past choices. The second is to make recommendations derived from the behaviour of other users through social, collaborative filtering, especially those who have displayed similar tastes and interests in the past. Both approaches make use of users' ratings of Internet documents, either given explicitly or derived implicitly from evidence of users' behaviour. More than simply developing specific implementations, however, the project aims to create an information filtering architecture that will afford the use of new techniques as and when they emerge.

Implementation of the base system is well under way and it is expected to be ready for testing and evaluation by user groups by the end of the year.

References

Avery, C. and Zeckhauser, R. Recommender systems for evaluating computer messages. Communications of the ACM, vol.40, no.3, March, 1997, p. 88-9.

Bouthors, V. and Dedie, O. Pharos, a Collaborative Infrastructure for Web Knowledge Sharing. Research Report RR-3679, 1999. INRIA.

Fåhræus, E. Intelligent Filtering on Usenet News: A User Study. Technical Report 97-003, Department of Computer and Systems Sciences, Stockholm University/KTH, 1997. http://www.dsv.su.se/~evafaahr/Filter.html

Glance, N., Arregui, D. and Dardenne, M. Knowledge Pump: community centered collaborative filtering. In Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering, Budapest, Hungary, 1998, p. 83-88. ERCIM Press.

Grudin, J. Why CSCW applications fail: problems in the design and evaluation of organisational interfaces. In Proceedings of CSCW'88, New York, 1988, p. 85-93. ACM Press.

Hill, W., Stead, L. Rosensteian, M. and Furnas, G. Recommending and Evaluating Choices in a Virtual Community of Use. In Proceedings of CHI'95, Denver, 1995. ACM Press.

Irmay, M. Some reflections about information filtering on the Internet, Swiss Federal Research Institute, 1997.

Joachims, T., Freitag, D. and Mitchell, T. WebWatcher: A Tour Guide for the World Wide Web. Research Report CMU-CS-xxx, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA, 1996.

Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L. and Riedl, J. GroupLens: Collaborative Filtering for Usenet News. Communications of the ACM, March, 1997, p. 77-87.

Lantz, A. How do experienced Users of the System Usenet News select their Information? Technical report, Department of Computer and Systems Sciences, Stockholm University/KTH, 1993.

Lantz, A. Useful Criteria for Intelligent Filtering? Technical report 95-042, Department of Computer and Systems Sciences, Stockholm University/KTH, 1995.

Lueg, C. Considering Collaborative Filtering as Groupware: Experiences and Lessons Learned. In Reimer, U. (Ed.) Proceedings of the Second International Conference on Practical Applications of Knowledge Management (PAKM'98), October, 1998.

Morita, M. and Shinoda, Y. Information Filtering Based on User Behavior Analysis and Best Match Text Retrieval. In Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94), 1994.

Procter, R. and McKinlay, A. Social Affordances and Implicit Ratings for Social Filtering on the Web. In Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering, Budapest, 1997. ERCIM Press, p. 89-96.

Resnick, P. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In Proceedings of CSCW'94, Chapel Hill, 1994, ACM Press, p. 175-186.

Resnick, P., Zeckhauser, R. and Avery, C. Roles for Electronic Brokers, Twenty-Second Annual Telecommunications Policy Research Conference, October, 1994.

Rucker, J. and Polanco, M. Personalized Navigation for the Web. Communications of the ACM, March, 1997. p. 73-75.

Sarwar, B.M., Konstan, J., Borchers, A., Herlocker, J., Miller, B. and Riedl, J. Using filtering agents to improve prediction quality in the GroupLens Research collaborative filtering system. In Proceedings CSCW'98, New York, 1998. p. 345-54. ACM Press.

Schmutzer. R. Scenarios for Voting and Rating Using Web4Groups. In R. Alton-Schiedl, R. Schmutzer, P. Sint and G. Tcsherteu (Eds.) Rating, Voting and Annotations. Oldenberg, 1997.

Stanyer, D. and Procter, R. Improving Web Usability with the Link Lens. In Mendelzon, A. et al. (Eds.), Journal of Computer Networks and ISDN Systems, Vol. 31, Proceedings of the Eighth International WWW Conference, Toronto, 1999. Elsevier, p. 455-66.

van Bommel, P., Koster, C. and van der Weide, Th. Profile – A Proactive Information Filter. Technical Note CSI-N9602, Computing Science Institute, University of Nijmegen, 1996.

Perception of pictures without graphical interface

Zdenek Mikovec, Pavel Slavik

Czech Technical University, Dept. of Computer Science and Engineering
Karlovo namesti 13, 121 35 Praha 2, Czech Republic
xmikovec@fel.cvut.cz, slavik@cs.felk.cvut.cz

Abstract. This paper deals with a problem of handling of graphical information in an environment with some limitation. The problem arises when the work with graphical information is limited or impossible at all. The limitations can be either on user side (disabilities) or on the side of a system (small screen in case of handheld devices). This means that it is necessary to apply some effective access to graphical information in such an environment. Textual information can be accessed without any problem and the solution is to work with textual description of the picture. Such a description should allow information filtering, as the structure of graphical information can be complicated. In order to describe a picture in textual form we will use a XML based grammar. Its use will allow us to automize creation of tools for creation of picture description and browsing. A system has been implemented that allows the user to access graphical information without graphical interaction. In the case of handheld devices the textual communication could serve as a preview in order to get closer to the area of interest in a picture. This area can be displayed by limited graphical means. The system has been successfully tested with a good response from potential users.

Keywords: picture description, mobile computing, grammar, XML, graphical interaction, blind users

1 Introduction

The use of pictures in information systems (IS) intensifies communication between the user and computer systems. Due to large flexibility of graphical information it is possible to use it in many applications. On the other hand there are situations when the work with graphical information is limited either by users (disabilities of various kinds) or by a system (small screen in case of handheld devices). In this cases it is necessary to solve effective access to graphical information using textual information without graphical interface. Our solution is based on working with textual description of the picture. Graphical information can have complicated structure and thus we have to be able to filter out unnecessary information using special information filters [9]. Common methods for describing pictures (e.g. alt-text, "D-Tags") are not suitable for description of complicated graphical information, because there is no way how efficiently browse in such a description (specially defining level of detail, filtering information). Our approach is trying to enable perception of more complex pictures. To automize the implementation of a system for manipulation with textual information we use a formal language for description of the structure of the picture description. XML [6] based grammar [1],[7],[8] is used as a tool for formal language definition. The use of XML as an international standard allows us to implement system that can be used in any environment that uses XML.

2 Creation and browsing picture description

The main goal of the specification of methodology of creating picture description is to keep complexity and specific structure of graphical information and allow later efficient browsing. Therefore we choose the object oriented approach. For definition of picture description we used XML [6] based grammar (see section 2.1.4).
Browsing in a picture description is in this work specified as a browsing in a tree (where nodes represent objects and edges represent relations among objects) and filtering out unnecessary information. The user browses in a picture by means of a statement in a special query language based on XML.

2.1 Creating a picture description

The meaning of a picture is specified both by the list of objects in the picture and by relations among these objects. These relations are either of a structural nature (sub-object, left, right etc.) or semantic nature (a person talks to another person). A picture description is then derived from a combination of both relation types.
Graphical information we would like to process consists of two parts:

- objects, their characteristic and behavior
- relations among objects
  - structural (e.g. one object is a part of another one, position, shape)
  - semantic (e.g. behavior, semantic relation).

The formalism used is suitable for pictures with good internal structure that could be expressed in formal way (mostly by means of a graph). The system is oriented for the use in web environment and therefore we will deal in further with pixel oriented pictures.

2.1.1 Objects, their characteristics and behavior

The problem of describing an object is that the description creator can describe the object in different ways. To eliminate differences in object description and speed up describing we have defined a standard for object behavior - basic categories of information that describe objects and their behavior. The browsing user then chooses what characteristics of objects he wants to get (e.g. position, shape, etc.).
There are two basic categories of object description:

- geometrical (position, shape, ...), material, color, weight, category, detail
- action (Behavior of object - "Boy is running", etc.)

2.1.2 Relations among objects

The relations among objects are standardized too:

- semantic - defines semantic relation (e.g. object "foyer" is connected with object "bedroom")
- hierarchical - defines hierarchical relation (e.g. object "bad" is in object "bedroom")
- group - defines group of objects without hierarchical relations

This basic set of categories covers the main descriptions and relations of objects. When describing some special picture there could arise the necessity to define new categories. The possibility to create new categories is incorporated in a design of the grammar of picture descriptions (see section 2.1.4).

2.1.3 Structural and semantic view

When analyzing creation and browsing of the picture description we found out that we could perceive the picture from different points of view, which means that final description could consist of different objects, different object descriptions and different relations. Two types of view were defined:

A. Structural view (see Figure 2) - focused on structural relations

- The goal of this view is to give the user a standard for picture description.
- Major relation is "is in" relation (full arrowhead lines). The objects are ordered into a hierarchical tree. The hierarchy means that the child object is a part of the parent object (e.g. object "bed" is sub-object of object "bedroom" - see Figure 2)
- There are only few of other relations.B. Semantic view (see Figure 3) - focused on semantic relations
- The goal of this view is to describe significant objects and relations only.
- It is not necessary to order objects into a hierarchical tree as in the structural view.
- This view is useful if the picture should be described from several totally different views (these views consist of different objects and relations). In our example (Figure 1) the semantic view (Figure 3) describes how the rooms of a flat are connected with each other.

2.1.4 Grammar of picture description

A suitable means for handling a complex structure of a picture is the use of a grammar [7],[8]. We will use the formal grammar in the sense of theory of formal languages where grammar is defined as a quadruple G=(N,T,R,S) [1], [8].

Our grammar is developed with the idea of generating a universal description language for representing any objects, their behavior and relations:

```
N = {picture, view,
object, description}
description
T...alphanumeric
Symbols
```

```
R:  picture ::=
  description+ view+
     view ::=
 object+
     object ::=
```

```
S = {picture}                          description+
                                          description ::=
                                        description*
```

```
(Symbol '+' means that the element could be repeated one or several
times. Symbol '*' means that the element could be repeated zero or
more times.)
```

`Picture` defines the name of picture. *View* defines the type of view. *Object* defines each object in the picture. *Description* describes object characteristics, behavior and relations among the object. All elements have similar attributes.

2.2 Browsing in a picture description

Browsing in a picture consists of two activities: *browsing* in a tree of objects and *filtering* information. Both activities are described by the user by means of statements in XML query language. Special information filters are used to filter out unnecessary information [9] in order to speed up a process of retrieval of required information. These filters can filter objects, object characteristics and relations among objects.

2.2.1 Information filtering

Users communicate with IS in three steps:

- information type selection (structural or semantic) - see section 2.1.3
- reduction of information - filter out inessential information (objects, descriptions of objects) by defining structural and semantic constraints
- browsing in reduced information using queries

When browsing a picture it is possible to switch dynamically between structural (see Figure 2) and semantic (see Figure 3) view of picture.

Object and relation filter

First method how to filter out unnecessary information is to choose the type of object description we want to see. While there are defined categories of object descriptions:

- geometrical (position, shape)
- material, color, weight
- category, action (connected with), etc.

the browsing user can choose which description categories he wants to see (e.g. color of objects). Applying such a filter will cut out descriptions and objects that do not match the chosen categories.

Second method how to filter out unnecessary information is to define relations between objects we want to see (see section 3). These filters can be combined in a logical expression (using logical operations AND, OR) or applied in a sequence. These filters defined by user are dynamically incorporated into the browser and immediately applied to the browsed description.

3 Example and testing

The example shown below could be considered as an example from the area of mobile computing. As a testing picture we will use scheme of flat (see Figure 1). At first we will analyze created picture descriptions (see Figure 2,3).

3.1 Creating description

We have constructed two types of description of Scheme of Flat (see Figure 1). The first one is structural description (see Figure 2). On the Figure 2 we can see two hierarchical levels of objects. On the first level there are the rooms the flat consist of and on the second level there are objets placed in each room. In general we can define more objects and more hierarchical levels, but for the sake of simplicity we will use two levels only.

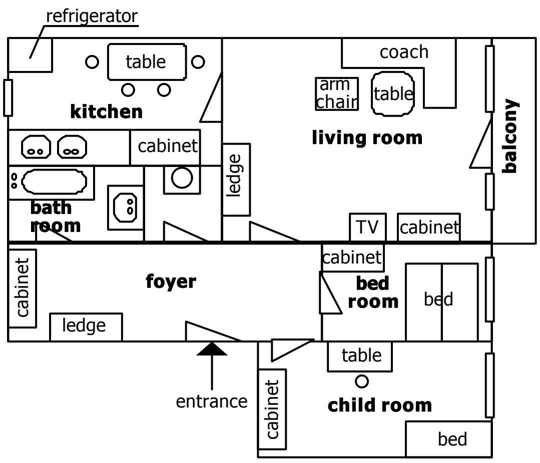


Fig. 1. Scheme of flat

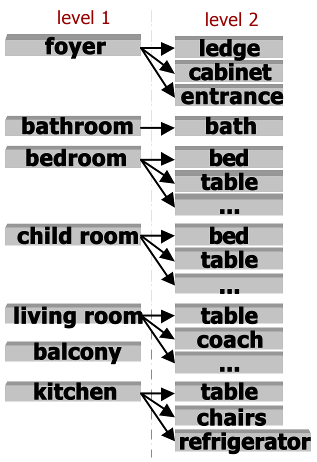


Fig. 2. Structural view

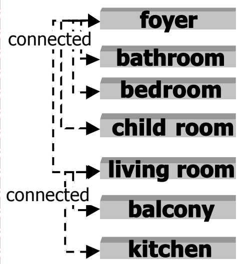


Fig. 3. Semantic view

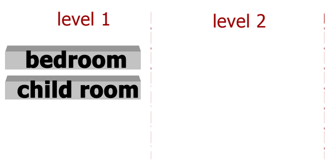


Fig. 4. Filtered objects - structural view

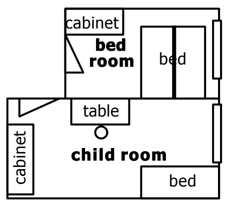


Fig. 5. Filtered objects - graphical view

The second description is called semantic one (see Figure 3). This description defines connections between rooms and will be used for finding way from one place to another one (see section 3.2). Similar relations can be defined between objects in the rooms. They can for example express objects of one suite (e.g. armchair, coach and table from the living room).

3.2 Browsing in description

For representation of objects and relations we use the following notation:

```
Object in the picture: living room
Filtered object: TABLE
Semantic relation: [connected to BEDROOM]
```

```
Hierarchical relation: living room
                          table
```

The following examples show possible tasks (a task that consists of a sequence of queries) when navigating in the flat. The queries have a form of statement in a special query language (e.g. XQL, XML-QL). These statements operate over the data structure of a picture described by means of a description language based on XML (see section 2.1.4). The statements in the following examples have only schematic form, because of the complicated structure of the statements of XML query language.

Task A: Find rooms with windows and bed.

1. Switch to structural view (Figure 2).
2. Set filter "window" AND "level 1"
("level 1" means: Show only objects on the first hierarchical level.)
Result (sub-graph of graph in Figure 2):

```
BEDROOM
CHILD ROOM
LIVING ROOM
KITCHEN
```

3. Set filter "bed" AND "level 1"
Result (sub-graph of graph in Figure 2) - see Figure 4:

```
BEDROOM
CHILD ROOM
```

4. Show filtered objects ("bedroom", "child room") - see Figure 5
The information obtained in "textual mode" can be displayed on a small screen (as we found a graphic area of interest).

Task B: How can I go from entrance to refrigerator.

We are locating entrance and refrigerator.
1. Switch to structural view (Figure 2).
2. Set filter "entrance" OR "refrigerator".
Result:

```
foyer
  ENTRANCE
kitchen
  REFRIGERATOR
```

We are finding the way from "foyer" to "kitchen".
3. Switch to semantic view (Figure 3).
4. Set filter "foyer"-"connected to"- "kitchen". (Is there a direct connection?)
Result: Nothing filtered. => There is no direct connection.
5. Set filter "foyer"-"connected to"-"any object"
Result:

```
BATHROOM [connected to FOYER]
BEDROOM [connected to FOYER]
```

```
CHILD ROOM [connected to FOYER]
LIVING ROOM [connected to FOYER] [connected to KITCHEN]
```

6. Set filter "filtered objects"-connected to"-"kitchen"
Result:

```
LIVING ROOM
```

The resultant path is: foyer -> living room -> kitchen

3.3 Implementation

The system consists of two modules:

- Description creator (module for picture description creation)
- Description browser (module for browsing in picture description without graphical interface)

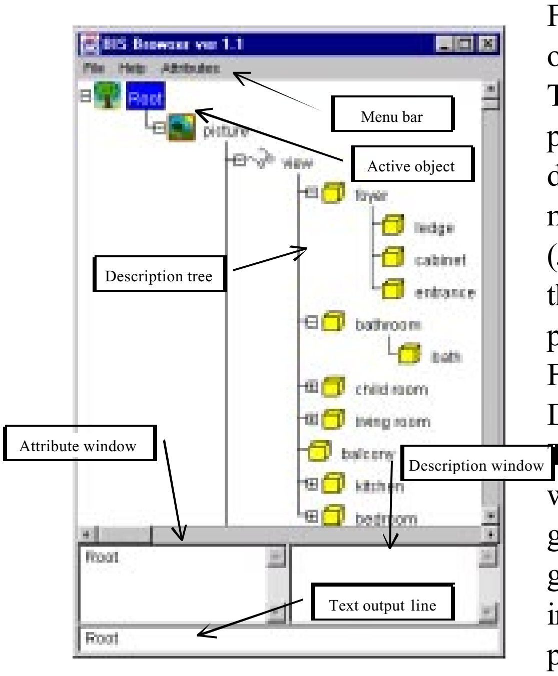


Fig. 6. Description browser User Interface

For detailed information about implementation of the system see [3],[4],[5].
The system has been implemented in JAVA [2] programming language and the data structure is defined using language based on XML [6]. This means that the system is platform independent (JAVA) and could be used in any environment that uses XML. The description is stored as a part of picture file (e.g. GIF, PNG). On the Figure 6 we can see user interface of Description browser.
The implementation has been intensively tested with group of blind and non-blind users. Both groups of users were asked to get some kind of graphical information when working with textual interface. The tests have shown that the perception of a graphical information without graphical interface does not caused any substantial problems to the users.

4 Conclusion and future work

We have defined methodology and also have implemented modules for the picture description creation and browsing the description. While this description contains structural and semantic information we can use special structural and semantic filters [9]. These filters allow us to automate "intelligent" searching in pictures. The use of XML [6] as the definition language for description file structure simplifies the implementation of our methodology in any application. The future work will deal with testing implemented modules on the platform of mobile computing. We are going to analyze special semantic relations between objects such as relative position of objects (object A is on the left side of object B, object in the neighborhood of object A). The system could be also adapted for the use of vector oriented pictures, where the methods for automatic description generation could be developed.

5 References

1. Melichar B.: Languages and Compilers (in Czech), 1996, CTU Prague

2. Chan Mark C., Griffith Steven W., Iasi Anthony F.: 1001 "Java Programmer's Tips", 1997, UNIS Publishing

3. Mikovec Z.: BSc thesis - Picture Description for Blind - Object Library, 1998, CTU Prague

4. Klima M.: BSc thesis - Picture browser for Blind, 1998, CTU Prague

5. Pavlica D.: BSc thesis - Creating Picture Description for Blind, 1998, CTU Prague

6. SGML/XML: http:// wwww.w3.org/XML/

7. Slavik P.: Syntactic Methods in Computer Graphics. Eurographics 83, North Holland, 1983, pp. 134-144

8. Slavik P.: Grammars and Rewriting Systems as Models for Graphical User Interfaces. In: Cognitive Systems. No.4-3, 1997, pp.381-399. ISSN0256-663X

9. Mikovec Z., Klima M., Slavik P.: Structural and semantic dialogue filters. In: Proceedings of the 2nd International Workshop Text, Speech and Dialog, Plzen, Czech Republic, September 1999, pp. 280-285

Ontology-Based Query Translation for Legislative Information Retrieval

**Kuldar Taveter, Aarno Lehtola, Kristiina Jaaranen,
Juha Sorva, Catherine Bounsaythip**

VTT Information Technology, P. O. Box 1201, FIN-02044 VTT, Finland
Phone: +358 9 456 6044. Fax: +358 9 456 6027.
Email: kuldar.taveter@vtt.fi

http://www.vtt.fi/tte

Abstract

The paper presents a conceptual approach that helps non-expert users to find legislative documents from textual databases. The concepts and inter-concept relationships of each legal domain involved are modeled by an ontology. Terms describing a concept at different levels of accuracy are attached to the concept, and a weight expressing the level of accuracy is associated with each of them. Initial queries presented by the user are matched against these terms. Thereafter the user is shown a graphical representation of the relevant subpart of the ontology that he can use for refining the query. The conceptual approach is preferred over traditional thesaurus because legal terms depend on the differences in law systems that can be expressed by ontologies.

1. Introduction

It is a basic right of citizens to have access to the laws and other legal texts that affect their life. The multi-national legislation of the European Union has made it even more important for ordinary citizens, as well as for professional law experts, to be able to read directives and other legal texts in their native languages.

Recently, it has become easier to get access to different legislative textual databases, but having access does not by itself guarantee that one is capable of finding the exact information needed for the situation at hand because in the countries of EU, directives and legal texts are stored in databases with different structures and in different languages. In addition to the EU legislation, every nation also has its own national laws stored in databases, sometimes even in multiple languages. Finnish laws, for example, are written in Finnish and Swedish as both are official languages in Finland.

Generally, two linguistic phenomena limit the ability of a user to choose effective search terms:

- polysemy (word having multiple meanings): reduces *precision* (the words used to match the relevant document may have other meanings that will be present in irrelevant documents found);
- synonymy (multiple words describing the same concept): reduces *recall* (word-based search needs to match the exact word form).

Legislation is a domain that has many special features deviating from normal everyday texts we read. The use of certain terms in legislative texts differs from general usage of these words, i.e. a word with a common meaning can also have a totally different meaning in the legal terminology. For example, in a law text the word ”consumer” may stand for ”the principal contractor”, among other possible meanings. When one wishes to find legal texts from different databases, it is thus not enough to use a general terminology to define query terms. Moreover, due to differences in law systems, legal terms are not always equal or even compatible in different languages, or even within one language when used in various contexts. For example, a prime minister is called ”premiärminister” or ”statsminister” depending on the country where these Swedish words are used.

Because of the issues mentioned above, some kind of help is necessary in defining query terms and finding the exact information needed for the situation at hand. The user needs not only a list of terms to search with, but also some kind of information about the way these words appear in the legislative texts and the meaning and function they have in them.

2. Representation of legal domains by ontologies

We make use of *ontology models* to give the user more information about legal terms. Ontology is a conceptual model of a problem domain. According to Guarino et al [1], **ontology** can be understood as an intensional semantic structure which encodes the implicit rules constraining the structure of a piece of reality. Ontologies are thus aimed at answering the question “What kinds of objects exist in one or another domain of the real world and how are they interrelated?”. Ontologies can be made explicit by forming a logical theory which gives an explicit and partial account of the above-mentioned intensional semantic structure. Such logical theory contains concepts, their definitions, and relationships between them like e.g. subsumption (inheritance) and aggregation.

We can also say that the use of ontologies in the context of our work is such an extension of the approach of controlled languages described e.g. in [2] where the means the language is controlled by is an ontology.

For specifying ontologies, we utilize the software tool CO(nceptual)N(etwork)E(ditor) [8] which has been developed at VTT Information Technology. Building an ontology for a legal domain with CONE involves:

- creating dictionaries of common concepts from the analysis of existing repositories (law texts) of the law system at hand;
- associating each concept with the terms in the language covered by the ontology;
- assigning to each term the weight expressing the level of accuracy of representing the concept;
- finding and expressing relations between the concepts.

The ontology models in CONE consist of *concepts*, *relations* between them, and *bridges* between the concepts in different models. An ontology model is designed to be used visually, i.e. the user can see a graphical representation of the model. To support the visual effect, concepts of different types are grouped semantically according to their functions.

Following the lines set in the paper [9], the basic concept types in our ontology models of law are *Actors* (humans or institutions), *LegalActs* (actions, activities) that they perform, *LegalInstruments* (documents) that they sign and thereby authorize, *Liabilities* (legal obligations) that they are liable for, and *MainConcepts* and *ConceptParts* that can respectively serve as patients and objects of LegalActs and LegalInstruments, and as consequences of LegalActs. In addition, there are concepts called *Headlines* that are important for visual grouping of the concepts. Headlines themselves are not actual concepts, they just give surface information in the graphical representation of a model. Please note that the type of a concept is often not absolute: the same concept can be seen as e.g. LegalAct or MainConcept, depending on the context and viewpoint. For instance, the concept ”repay” belongs to the type MainConcept in Figure 1, but it could also belong to the type LegalAct.

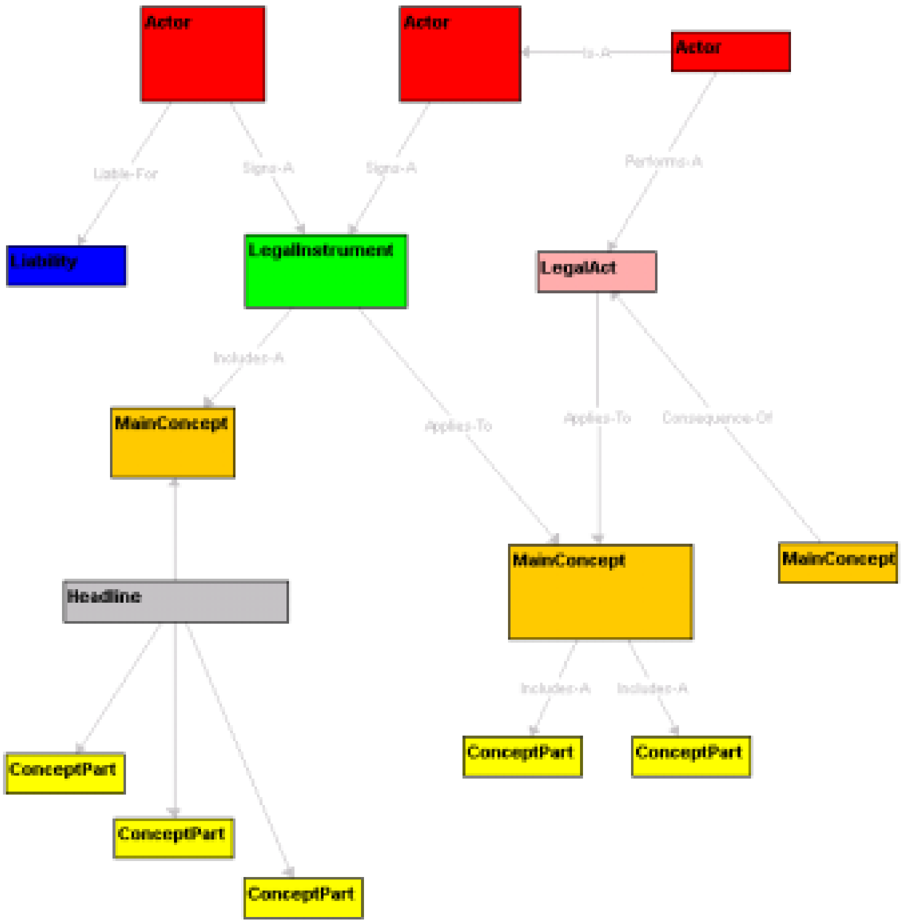


Figure 1. The organization of concepts in the ontology on a travel package law

In Figure 1 the types of concepts and relations in the ontology describing a package travel law is depicted. As it can be seen, concepts of different types are presented with different colors. The figure reveals that a legal ontology is not a traditional tree-formatted hierarchy. A tree representation would require unambigous subsumption relations which are not sufficient enough to describe the complexity of a law in a way that is also understandable to an ordinary citizen.

The inter-concept *relations* shown in Figure 1 reflect semantical relationships between different law concepts. The types of relations used in our legislative ontologies and their meanings are given in Table 1. In the graphical representations of the ontologies, general relations of the type ”Connects” are depicted as arrows without any type labels.

Concepts of different types are divided into clusters according to their functions. For example, the following clusters are present in Figure 4, which is a snapshot of the user interface of our system: ”Acts”, ”Consequences”, ”Consumer”, ”Organiser”, ”Package”, ”Headlines”, and ”Information”.

Each concept of the ontology is represented by natural language expressions called *terms* which can be single words or longer surface expressions. In a concept box of the graphical representation, there is a list of terms from very accurate ones to more general forms, for example the expressions ”package travel contract”, ”package contract”, and ”contract”, assuming that all the expressions do appear in the actual law texts. In the following a concept is referred to by its first term, because it is used most frequently to represent the concept and is therefore at the top of the list of terms of the concept.

According to our approach, each term pertaining to a concept is assigned the value between 0 and 1 expressing the level of accuracy the term represents the concept with. This enables the use of fuzzy logic described e.g. in [3] for matching query terms against concepts. For example, the term ”package travel contract” representing the concept of the same name is assigned the value 1.0, and the terms ”package contract” and ”contract” pertaining to the same concept are assigned the values 0.7 and 0.3, respectively. The use of accuracy values is explained in section 3.

RELATION	MEANING
Connects	General (unspecified) relation
Is-A	Subsumption
Includes-A	Aggregation
Performs-A	Actor → Legal Act
Signs-A	Actor → Legal Instrument
Applies-To	Legal Act → Patient, Legal Instrument → Object
Consequence-Of	Product of the Legal Act → Legal Act
Agent-Of	Representative of the Actor → Actor
Role-Of	Role of the Actor → Actor

Table 1. The types of relations and their meanings

The links between different ontological law models are organized with *bridges* that lead from one ontology model to another and thus guide the translation between the models and languages. There can be a bridge between any two concepts in two different ontology models. Usually there exists a bridge between concepts that are translational equivalents in two languages and/or share the same function in different laws. According to [4], if a concept does not have any kind of counterpart in another ontology model, it can be linked to a superclass concept in the target model. For example, since the concept ”varallisuusvahinko” (damage done to one’s property) in Figure 2 has no equivalent concept in the target model, it can be connected to the concept ”damage” at the superclass level. Even if the bridge does not describe an equivalent translation, the user can understand the connections between the concepts by visual comparison of the graphical representations of the original ontology model and the target model.

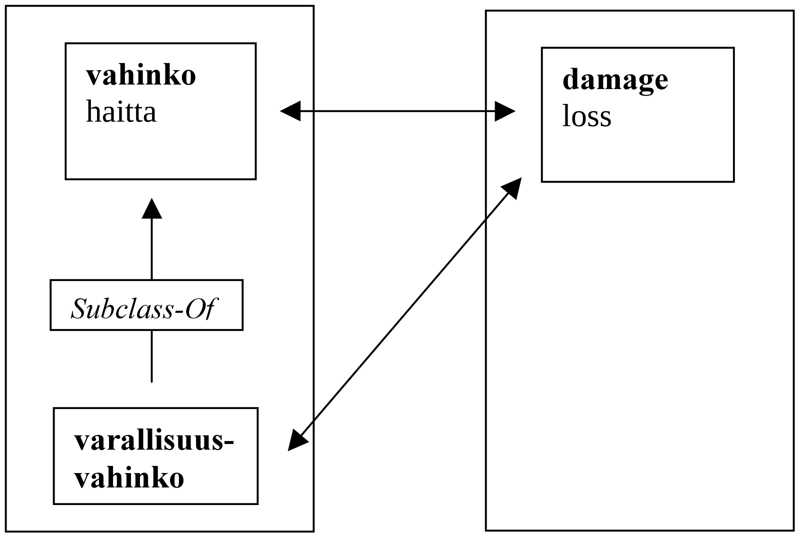
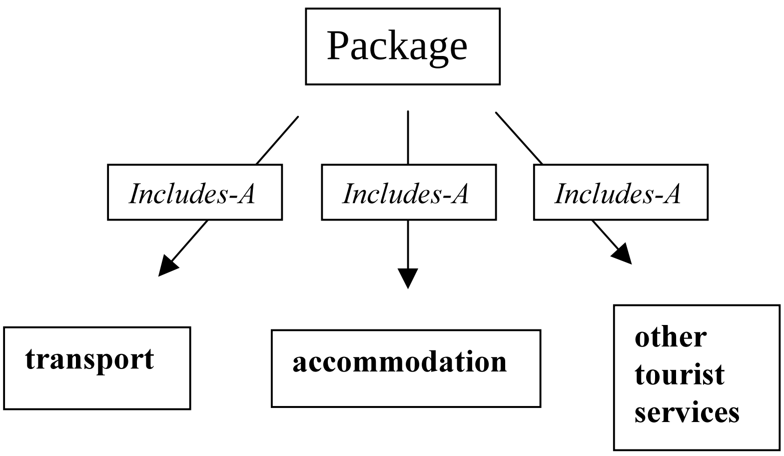


Figure 2. Formation of bridges between concepts of different ontologies

Let us take a simple example of conceptual modeling of legislation by an ontology. The package travel law of EU defines the concept ”package” as follows:

"package" means the pre-arranged combination of not fewer than two of the following when sold or offered for sale at an inclusive price and when the service covers a period of more than twenty-four hours or includes overnight accommodation:
(a) transport;
(b) accommodation;
(c) other tourist services not ancillary to transport or accommodation and accounting for a significant proportion of the package.

A piece of an ontology model based on the previous law text could be:



The concept ”package” consists of different parts which are represented in the model above. The relation between the concepts is ”Includes-A” (aggregation). Other terms referring to the concept ”package” are then added, and the values describing the levels of accuracy of representing the concept are assigned to them. After that the concept could look like this:

Package
package: 1.0
package travel: 0.9
package holiday: 0.7
package tour: 0.5
inclusive tour: 0.3

The concept ”package” is then linked to a comparable concept in a model describing another law system in another language (for example to the Swedish model):

Package		**Paketresa**
package: 1.0 package travel: 0.9 package holiday: 0.7 package tour: 0.5 inclusive tour: 0.3	↔	paketresa: 1.0 resepaket: 0.9 semesterpaket: 0.7 paketarrangemang: 0.5

Please note that the number of terms referring to the same concept can vary in different languages. Please note also that the expressive power of our present model does not enable to represent the *rule* present in the definition of the legal term ”package”, according to which ”package” means the combination of *at least two* of the following: transport, accommodation, and other tourist services. Expressing of this rule is, however, important for expressing legislative norms *per se*, but not to translating between the concepts of different law systems expressed as ontologies.

3. Ontology-based query matching and translation

Our system is embedded into an interface through which the user can make queries in one language to search for legislative texts from different EU and national databases. By using that interface, the user selects the source and target languages and systems of law (e.g. EU or national). Based on them, the source and target ontologies are determined. The first step in the ontology-based query translation is matching query terms against terms representing concepts in the source ontology in order to determine the concept most probably relevant to the query. The values expressing the levels of accuracy of terms in the source ontology are used for that. For example, let us assume that the source ontology is an ontology about the EU legislation on package travels expressed in English. If the query term used is ”supplier”, it can be matched with two concepts of the ontology: ”retailer” and ”organizer”, because they both include ”supplier” as one of their terms[1]. In order to decide between the two alternatives, accuracy values and fuzzy logic are used as depicted in Figure 3. Since the accuracy values representing the degrees with which the term ”supplier” represents the concepts ”retailer” and ”organizer” are respectively 0.6 and 0.2, it can be concluded that the degree of membership of ”supplier” in the concept ”retailer” (60%) is higher than the degree of membership of ”supplier” in the concept ”organizer” (20%), and the concept ”retailer” can be assumed to be more relevant than the other one. The number of alternatives to decide between can naturally be even higher than two.

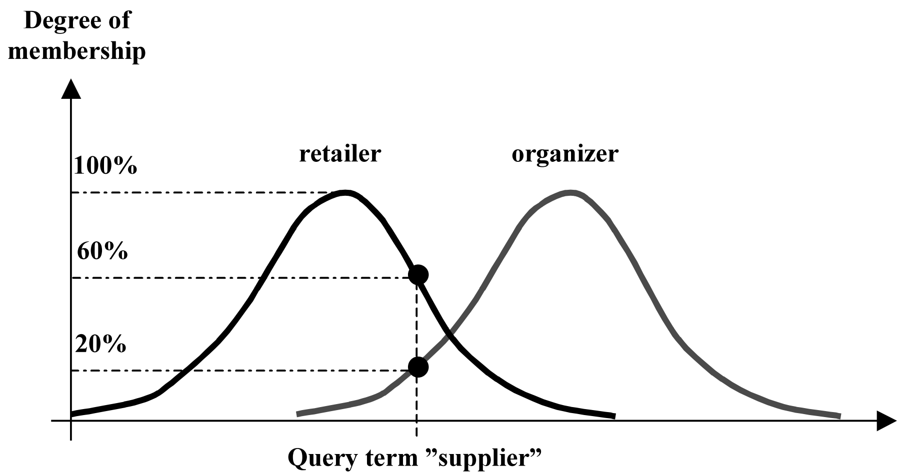


Figure 3. Comparing the term's degrees of membership in two concepts

At the next step of ontology-based query translation, the source ontology is represented graphically with the found most relevant concept highlighted. The described situation is depicted in Figure 3 as a snapshot of the user interface of our system. In the upper rightmost window of the figure, the English terms of the EU legislation on package travels representing the concepts initially found from the query or selected by the user are presented. The lower window in the right shows the corresponding terms in the legal terminology of the Finnish national law on package travels. The middle window depicts all the terms of the Finnish national law on package travels. From there the user proficient in Finnish can directly select the terms he is interested in, thus skipping the phase of browsing the ontology. The translation between the laws of EU and Finland is performed by using the ”bridge” relationships (cf. Section 2) between the concepts of ontologies about EU's and Finnish legislation on package travels. The user can view the latter ontology by pushing the button ”Display target model” depicted in the figure.

Since the source ontology also presents other concepts related to the original target of the search, the user can *expand* the search by choosing other related concepts in the ontology model. For example, in Figure 4 the user has also selected the concept ”local agencies” which is linked to the concept ”retailer” by the relation of the ”Agent-Of” type, as a result of which the translated query will be expanded with the terms representing the concept ”local agencies” in the target language, i.e. in Finnish. The accuracy values can also be used for weighing translated queries by assigning each

---

[1] A term representing the concept ”organizer” is actually ”service supplier”, but according to the dissertation [7], basic words of phrases like e.g. ”supplier” of ”service supplier” can be used as terms with reduced weights (accuracy values)

component term of a translated query a weight proportional to its accuracy value in the target ontology. Moreover, each relation type can also be assigned a similar accuracy value representing the strength or reliabilty of the relation. For example, the strengths of the ”Is-A” and ”Applies-To” relations could be 0.8 and 0.5, respectively. These values can be used for additional expansion and weighing of search terms. A relevant methodology is described in the work [7]. Consequently, our approach also provides *metadata* that can be used for weighed query expansion in accessing legislative textual databases. However, at present our system is not used for that.

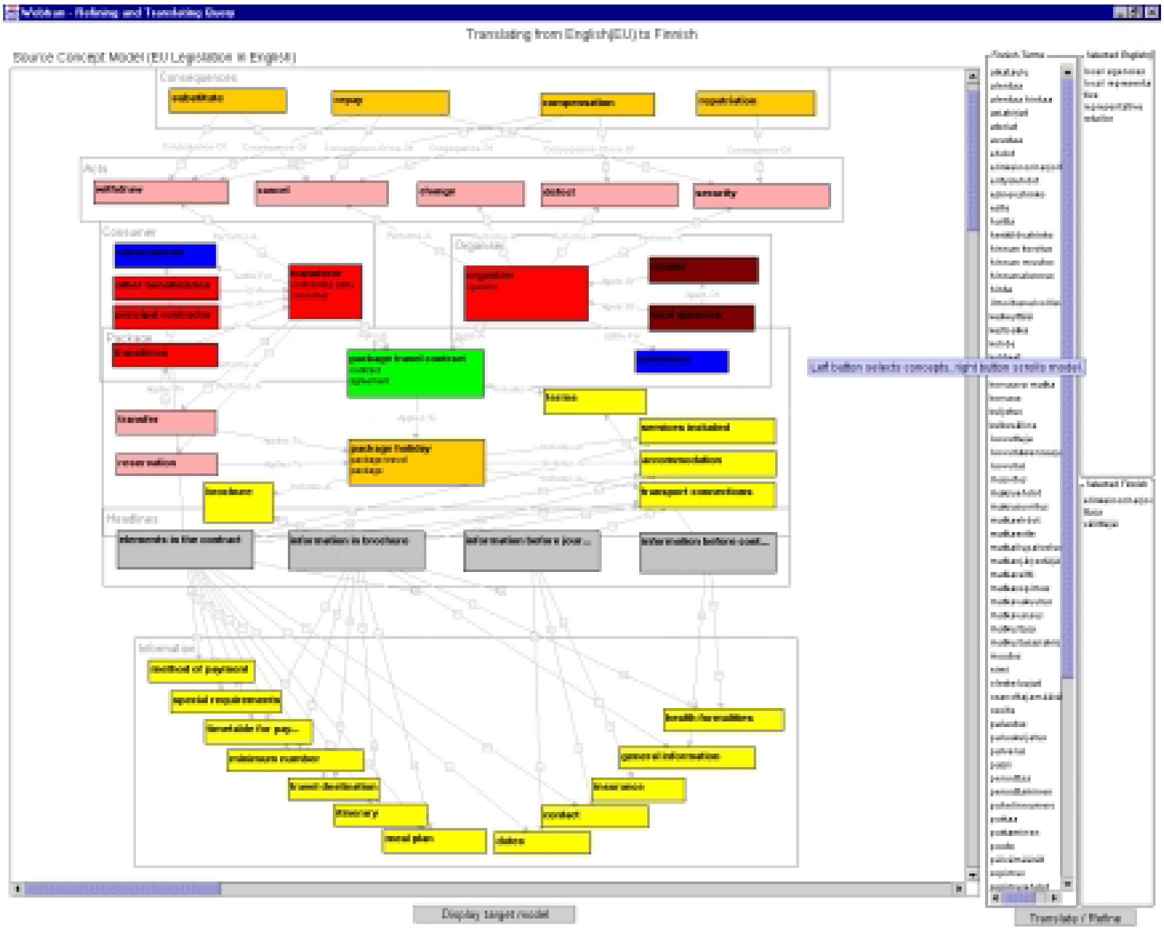


Figure 4. Ontology-based translation of terms from the EU legislation on package travels expressed in English to the Finnish national legislation on package travels

The law systems and languages that can be translated between with the help of our system are currently:

- the legislation of EU in English, Finnish, Swedish, and French;
- the national legislations of United Kingdom, Finland (in Finnish and Swedish), Sweden, and France.

Presently the only subarea of law that is supported is the legislation on package travels, but other subareas will be added in the near future.

Our system has been implemented as a Java applet which has an interface to the search engines connected to legislative databases. The query expressions, as well as the choices of the language and law system by the user, are passed to the applet through HTML applet parameter tags as character strings.

The applet connects to the serverside or local relational database of ontology models, loads the relevant models, and displays them to the user within the WWW browser that is used to view the applet.

4. Related work

Our method of representing ontologies is based on *conceptual graphs* which is a knowledge representation language introduced by John F. Sowa [12]. Conceptual graphs use graphical representation as a method of encoding knowledge. In a conceptual graph, concept nodes are used to represent entities, attributes, states, and events, while relation nodes are used to show how these concepts are related to each other. One of the main differences between our ontology models and conceptual graphs is that in our models attributes of a concept are represented as properties of the concept, while in conceptual graphs attributes are represented as concept nodes.

The Hyperlaw system described in [5] is based on an information retrieval system model called EXPLICIT. This model uses a two-level structure to make the different parts constituting the body of data explicit at the document level, called the hyperdocument, and at the auxiliary data level, that is, at the level of the semantic structure according to which the indexing terms are organized, called the hyperconcept.

The two levels are linked by the relations between the concepts and the documents the concepts describe. At the same time the single elements contained on each of the two levels are interconnected: the documents are linked by references or citations, while the links among the auxiliary data are made up of the semantic structure in which the terms are placed.

The EXPLICIT model makes it possible to display the conceptual structure of the indexing terms, so that the proper semantic context in which each term is placed becomes known according to the meaning it is given in the indexing phase. This semantic structure can, at all times, be actively searched by the user, who, therefore is able effectively and immediately to enhance his query.

The model makes use of a conceptual framework tailored for a specific application domain and makes that scheme available for active utilization by the user during his interaction with the system, thus providing a reference structure for the process of query formulation. Another feature is concurrent use of different conceptual schemes for the same application domain.

The FOOD system described in [6] proposes a Fuzzy Object Oriented Data Model representing both vague and uncertain information by means of linguistic qualifiers. The interpretation proposed for uncertainty qualifiers is based on the assumption that they contain an implicit twofold information: they both specify that the declared (vague) attribute value can be violated by the actual value to a given degree, and impose an imperative safeguard constraint on this possible violation.

5. Future work

The present version of our system concentrates only on conceptualizing and visualizing package travel law, a very specific subdomain of legislation. In the future, as more models of other subdomains will be created, we will be faced with the problem of bringing together these parts to form an easy-to-use system for searching the ontology models pertaining to many fields of legislation.

Obviously, as the number of legislative fields increases, the ontology models will fast become much too numerous to be incorporated into a single, unified model. Moreover, many of the legal terms involved might carry different connotations (and thus have different conceptual connections) in different subdomains, resulting in ambiguity, and possibly considerable confusion for the end user. Therefore hierarchical organization of the legislative fields is necessary in order to isolate the various subdomains and provide the user with a sensible method for accessing the correct one.

In the hierarchical domain system, the ontology models would be divided into several layers, which branch into the layer beneath them. The top level would include the very basics of legislation – the concepts that describe the law-making process in general and define the major subdomains of the legal system in question (e.g. customer protection law, criminal law, etc.). Each of these subdomains would in turn contain concepts corresponding to sub-subdomains, which the user could select and view. For example, the user could select the concept "customer protection law" at the top level and then move on down to "travel law" and finally to "package travel law", if he wanted to find the very basic concepts involved in these specific laws.

In the later versions of our system, the end user would be able to navigate graphically through the described layers. The concepts in the top-level model would correspond to the subdomains of the top ontology model, and by selecting the desired concept the user would be able to zoom in and view the second layer of the hierarchy, with more detailed concepts. At the bottom level, the user could inspect the basic concepts and terms of one particular subdomain (e.g. legislation on package travels).

There would also be implemented an automatic search system for the ontology models of the various subdomains, which would, if wished, automatically direct the user to the relevant subdomain, based on the query submitted by the user. In case of ambiguities (i.e. the same term is used in different contexts in different subdomains), fuzzy matching would be used. As another alternative, the user would be taken to the lowest possible level containing all the subdomains in question. For example, if the user made a query with a term used both in the legislation on package travels and product marketing, he would be shown the "customer protection" ontology model with the concepts "package travel law" and "product marketing law" highlighted.

As another important issue to be treated in our future work, a well-defined methodology for capturing the concepts, relations between them, and the terms representing the concepts should be worked out. The methdology should also provide guidelines for assigning accuracy values to terms and possibly also to relations, as this is presently done just by intuition.

Our approach can also be applied in domains other than legislative ones. The use of a similar methodology in the domain of foreign trade is described in the paper [4].

Our work should also be placed into the framework of task/context analysis as described e.g. in the papers [10] and [11].

It is also worthwile to consider how could new terms be included at run time, so that the ontology model and the query at hand would adapt themselves to these terms.

6. Conclusions

The *main contribution* of our approach is that instead of using a general language in a search for legislative documents, the user can take advantage of the legal terms provided by the ontology model. The user starts out with general-language or specific query terms that lead to certain concept(s) in the source ontology representing the legal system and language selected by the user. As another novel feature of our approach, *fuzzy logic* is used for matching query terms against terms representing concepts in the source ontology in order to determine the concept(s) relevant to the query.

In the source ontology, the user can specify the *exact concepts* to be used in the query to the databases. Since the ontology also presents other concepts related to the original goal of the search, the user can *expand the search* by choosing related concepts in the ontology model. The terms representing the selected concepts are then translated according to the links between the source ontology and the ontology model of the target law system and language.

Our approach also provides *metadata* that can be used for *automatical expansion of translated queries and weighing of their constituent query terms* in accessing legislative textual databases.

The ontology models can also be of advantage in the actual legislative process, for example when *harmonizing the terminology* of legal areas between different countries and languages. This could offer considerable benefits for example in applying the EU directives to national laws. Even the information structures of law texts could be harmonized through the use of the ontology models describing the contents of the laws.

The harmonizing process could be based on the *intralingual translation* for example between the terminology used in the British legislation and the terminology of the EU-directives in English. During the design of the ontologies for our system it was noticed that the terminology in Finnish had also considerable discrepancies between the national and EU legislation. The ontology models can offer the user information about the terminology in one specific language used in different legal sources.

7. Acknowledgements

The described work is mainly funded by the Technical Development Centre of Finland (TEKES). The authors are very grateful to TEKES and to the industrial partner of the project TietoEnator Group from Finland.

8. References

1. Guarino, N., Giaretta, P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In: N. J. I. Mars (ed.), Towards Very Large Knowledge Bases, IOS Press 1995, pp. 25-32.
2. Lehtola, A., Tenni, J., Bounsaythip, C., Jaaranen, K. WEBTRAN: A Controlled Language Machine Translation System for Building Multilingual Services on Internet. Machine Translation Summit VII '99 (MT Summit 99), Sep. 13-17, 1999, Singapore, 9 p.
3. Cox, E. The Fuzzy Systems Handbook. 2nd ed., AP Professional, 1998.
4. Taveter, K. Intelligent Information Retrieval Based on Interconnected Concepts and Classes of Retrieval Domains, Proceedings of the Eigth DELOS Workshop User Interface in Digital Libraries, Stockholm, Sweden, 21-23 October, pp.39-43.
5. Agosti M., Colotti R., Gradenigo G. A Two-level Hypertext Retrieval Model for Legal Data. In: A. Bookstein, Y. Chiaramella, G. Salton and V.V. Raghavan (eds.), Proceedings of the 14th ACM-SIGIR International Conference on Research and Development in Information Retrieval, Chicago, USA, 1991, 316-325.
6. Bordogna, G., Pasi, G. Linguistic Qualifiers of Vagueness and Uncertainty in a Fuzzy Object Oriented Data Model, Proceedings of the Third International ICSC Symposia on Intelligent Industrial Automation (IIA'99) and Soft Computing (SOCO'99), Genova, Italy, June 1–4, 1999, pp. 719-725.
7. Kekäläinen, J. The Effects of Query Complexity, Expansion and Structure on Retrieval Performance in Probabilistic Text Retrieval. Academic Dissertation, University of Tampere, Department of Information Studies, Finland.
8. Kankaanpää, T. Design and Implementation of a Conceptual Network and Ontology Editor. VTT Information Technology, Research Report TTE1-4-99, June 1999, 74 p.
9. Hafner, C. Representation of knowledge in a legal information retrieval system. In: Information Retrieval Research, R. Oddy, S. Robertson, C. van Rijsbergen, P. Williams (Eds.), Buttersworth, London, 1981.
10. Holtzblatt, K., Beyer, H. Contextual Design: Principles and Practice. In: Field Methods for Software and Systems Design, D. Wixon and J. Ramey (Eds.), John Wiley & Sons, Inc., New York, 1998.
11. Richardson, J., Ormerod, T. C., Shepherd, A. The role of task analysis in capturing requirements for interface design. Interacting with Computers 9(1998), pp. 367 – 384.
12. Sowa, J. F. Conceptual Structures, Information Processing in Mind and Machine. Addison-Wesley Publishing Company, 481 p.

User Testing in Industry: A Case Study of Laboratory, Workshop, and Field Tests

Morten Hertzum
Centre for Human-Machine Interaction
Risø National Laboratory, Denmark
morten.hertzum@risoe.dk

Abstract. Applied user testing involves more usability evaluation methods than laboratory tests and is critically dependent upon a number of issues seldom treated in the literature. The development of the system described in this longitudinal, diary-based study evolved around five user tests: a laboratory test, a workshop test, and three field tests. The user tests had a substantial impact on the focus of the entire development effort in that 25% of the primary developer's time was spent solving problems encountered during the tests. The laboratory test made use of set tasks and was biased toward *how* tasks were performed with the system, at the expense of *what* tasks could be performed. The workshop test was more informal and apparently led the users to adopt a more exploratory attitude. Careful arousal and management of the users' commitment to participate actively proved essential to effective user testing, especially during the field tests.

1 Introduction

Slightly caricatured the literature depicts user testing as videotaped usability laboratory tests with set tasks and no context. However, accounts of how user testing is done in practice evidence that applied user testing takes many forms to meet real-life needs and limitations (see, e.g., Brooks, 1994; Szczur, 1994; Zirkler & Ballman, 1994). This study provides field data on the user testing done in a project concerning the development of a graphical front end for an existing application. The purpose is to investigate the effectiveness of certain, very different user testing methods with respect to how well they fit into the development process and how well the tests resemble the use context that the evaluated system is intended to support.

The data collected for this study are a diary that covers the activities of the primary systems developer and the reports from the five user tests the project went through on its way from early prototype to release of the system: a laboratory test, a workshop test, and three field tests. The laboratory test was a conventional thinking-aloud study, except that the evaluators pinpointed misconceptions and other problems on the fly rather than by analysing videotapes. Unlike the laboratory test the remaining user tests emphasised low cost and an informal atmosphere. The workshop test, conducted by the developers in a conference room, consisted in having a group of users work two by two without being closely observed. This bears some resemblance to co-operative evaluation (Wright & Monk, 1991), where designers serve as evaluators of their own systems, and to constructive interaction, a variation of the thinking-aloud study where two users jointly discover how to use a system by trying it out (O'Malley et al., 1984). The field tests were similar to beta tests, hence they were performed by the users without supervision. Smilowitz et al. (1994) find that beta tests may be a cost-effective usability evaluation method.

The effectiveness of a user test depends on a number of interrelated issues. This study compares and contrasts the investigated user testing methods along four dimensions of test effectiveness. This four-dimensional framework is outlined in the next section. Section 3 through 5 describe the studied project, the data collection, and the user tests. The sixth section discusses the effectiveness of the tests; and the seventh section summarises the lessons learned.

2 The Effectiveness of User Testing

Several studies indicate that user testing is currently more effective than other competing approaches to the development of systems that meet the users' needs (Brooks, 1994; Pejtersen & Rasmussen, 1997). This does however not mean that any kind of user testing fits any situation equally well. User testing methods can for example be divided into types according to the approximate point in the development process at which the issues addressed by the test match the major design concerns (Rubin, 1994) or according to the aspect of the use situation at which the test is focused (Pejtersen & Rasmussen, 1997).

The high-level framework applied in this study (see Figure 1) reflects that effective user testing is dependent upon a good fit between the test and the rest of the development process as well as between the test and the use context. The framework identifies four key dimensions of test effectiveness and offers a simple way of illustrating how a user testing method balances these dimensions. Applied user testing is subject to a number of trade-offs, for example between consumed resources and obtained benefits in terms of impact, robustness, and ecological validity as well as between ecological validity and the control required to achieve robustness.

2.1 Ecological Validity

The situations in which user tests are performed differ from the real world in that some aspects of the real world have been left out of the test situation while other aspects that do not exist in the real world may have been added. The closer the test situation is to the real world, the more ecological the test. Ecological gaps between the test situation and the real world introduce a risk that what appears as a problem during a test will not be a problem during actual, real-world use and that some of the problems that will surface during actual use will not surface during a test. While a number of studies compare various usability evaluation methods with laboratory tests, hardly any studies compare evaluations with actual, real-world use of the systems. Thus little is known about to what extent the problems detected during tests are ecologically valid. In a notable study Bailey et al. (1992) found that only two of the 29 problems encountered during a heuristic evaluation had an impact on the users' task completion times and subjective preference.

Thomas & Kellogg (1989) identify four areas of ecological gaps in laboratory tests: *User gaps* are caused by individual differences between users and by a gap between the users' motivation to perform in the laboratory and in their day-to-day work. *Task gaps* are caused by difficulties in generalising from the tasks that can be observed in the laboratory to all the tasks the users

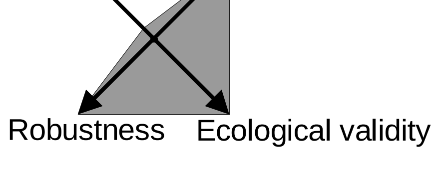


Figure 1. The effectiveness of user testing. The shaded area in the figure indicates the ideal user test that requires a minimum of resources and yields full impact, robustness, and ecological validity.

will want to carry out with the system as well as by differences between a short list of well-defined tasks to be performed in the laboratory and the real world's ongoing stream of possibly ill-defined tasks. *Artefact gaps* are caused by differences between using a single system in the laboratory and using a larger computing environment in the real world and by differences between short-term use in the laboratory and long-term use in the real world. *Work-context gaps* are caused by differences in job context, social context, and cultural context. The way to get around these gaps is to go to the field (e.g., Whiteside et al., 1988), but doing so reduces the robustness of the test.

2.2 Robustness

A method is robust when it produces fairly stable results across a range of minor variations in the test situation. This means that a rerun of the test will yield essentially the same results. The robustness of a user test depends on a number of issues that are within the evaluators' control, such as the number of participating users and the level of detail at which it is prescribed what goes on during the test sessions. However, user tests are also affected by a number of issues that are beyond the evaluators' control and thus vary from one instance of the test to the next. One such issue is the users' reaction to the often stressful test situation. A typical laboratory test involves trying to use a new system, being videotaped, and performing in front of others. Each of these three circumstances is experienced as unpleasant by many people and their combination creates a situation that is stressful to most people (Schrier, 1992).

A user testing method must be both robust and ecologically valid to reliably predict the parts of a system that need to be changed because they are confusing, slow users down, or do not match the users' needs. Whereas it is a primary intention of laboratory tests to provide a controlled environment where various sources of variability can be kept to a minimum, the resulting robustness is achieved at the expense of reduced ecological validity. Many dimensions of laboratory tests have been investigated including the sufficient number of users (e.g., Lewis, 1994), individual versus co-operating users (Hackman & Biers, 1992), the level of experimenter intervention (Held & Biers, 1992), and various methodological pitfalls (Holleran, 1991). Recently, Jacobsen et al. (1998) have shown that laboratory tests are subject to a considerable evaluator effect in that different evaluators, who analyse the same test sessions, detect markedly different sets of problems. Field tests refrain from tight control of the test situation and thus achieve their higher ecological validity at the expense of reduced robustness.

2.3 Impact

The impact of a user test is its ability to bring about changes in the evaluated system or the development process. That is, the impact concerns the persuasive power of the test rather than its predictive power, i.e. its ability to predict what aspects of the system that will cause problems to users during real-world use. The impact of a user test is most directly seen in relation to the development team but a user test may also interact with other actors in the development organisation and its outcome may therefore also have an impact on, for example, management or marketing (see Brooks, 1994; Zirkler & Ballman, 1994).

Whenever a problem predictive of actual use is left unaddressed an opportunity to improve the evaluated system is missed and the effort that went into finding the problem is wasted. However, the time required to fix a problem must be weighted against the benefit of fixing it and the benefit of spending the same amount of resources on any other outstanding task. Sawyer et al. (1996) define the impact ratio of a usability evaluation as the number of solved problems divided by the total number of problems found, expressed as a percentage. On the one hand, this way of calculating impact incorrectly assumes that all problems are equally severe and that problems are either left unaddressed or solved completely. On the other hand, the impact ratio is easy to understand and calculate, and it provides a rough measure of the action taken in response to a usability evaluation. Sawyer et al. (1996) report an impact ratio of 78% averaged over ten usability inspections, but their calculations are based on the developers' commitment to fix a certain number of problems so the achieved impact ratio may be lower. Whiteside et al. (1988) tested multiple versions of a single system and report an impact ratio of 65% for the early, in-house tests and 48% for the subsequent field tests.

2.4 Resources

Several studies provide formulas to estimate the costs of conducting user tests and try to justify these costs by converting the estimated benefits of performing the tests into cost savings (see, e.g., Bias & Mayhew, 1994). However, despite logical arguments to the contrary the subjective experience of many developers is that usability work lengthens the projects, adds expenses, and fails to prevent that new problems show up when the systems are released for actual use (Lund, 1997). As a result practitioners tend to be cautious and show a strong preference for methods that are low-cost in terms of the time, expertise, and equipment required to apply them. For example, Rowley (1994) reports how a mobile usability testing facility can provide a low-cost alternative to a dedicated usability laboratory. Nielsen (1993) advocates that though low-cost, discount evaluations are inferior to expensive, deluxe evaluations, discount evaluations are highly cost-effective and vastly superior to doing no evaluation work at all.

The choice of user testing method directly affects the cost of finding problems in a system, but the total cost of user testing also includes the cost of addressing the detected problems. This cost may be affected by how well the test fits into the development process but apart from that the cost of addressing the problems is not affected by the choice of user testing method. If the total cost of user testing is dominated by the cost of addressing the problems, the amount of resources that goes into the conduct of a test becomes less critical. Unfortunately, little is known about how the cost of conducting user tests compares with that of addressing the detected problems.

3 The F&N Project

The company where the case study took place is a publicly owned software house with around 2100 employees. The investigated project, initiated in December 1994 and completed in January 1996, consisted in the development of a graphical user interface for the Filing and Notification (F&N) system which contains information about citizens for use by local authorities. Most importantly, the F&N system gives a complete overview of the business that the local authorities have with a citizen in terms of business files which deal with tax, social security, kindergarten, or school. The F&N system was developed in the 1970's as a mainframe application and is used daily by several thousand civil servants in the Danish municipalities. However, efficient use of its character-based user interface requires dedicated training, regular use, and an extensive printed manual.

The purpose of the F&N project was to make a Windows version of the F&N system. To minimise the development time and cost of this version it was decided to implement it as a graphical front end on top of the existing mainframe application. Thus, the information displayed in the front end was to be obtained from the mainframe application, which would run in the background, and data entered into the front end were to be transferred to the mainframe application for updating the central database. This approach preserves the investment in the mainframe application and provides a way to gradually migrate legacy systems to a client/server environment. Since the project concerned the development of a new front end for an existing system, the task consisted chiefly in the design and implementation of a user interface. The amount of analysis was comparatively smaller.

The F&N project group consisted of a project manager, a primary systems developer who was the mainstay of the project, a secondary systems developer who was responsible for the development of a module for handling free-text notes, and an online help writer. The primary systems developer was identical to the present author who was at that time employed in the organisation where the study took place.

4 The Collected Data

The data collected from the project were the reports from the user tests and a diary that covered the activities of the primary systems developer. These data were supplemented with this author's first-hand knowledge of the project.

4.1 Problems

In classifying the problems encountered during the user tests the present study distinguished between utility, defined as ”the question of whether the functionality of the system in principle can do what is needed”, and usability[1], defined as ”the question of how well users can use that functionality” (Nielsen, 1993). The following categories were used to classify the problems: (1) *Utility problems*. A facility or piece of information needed by the users was not present in the system. Example: A citizen's marital status is a prominent piece of information because it determines whether the citizen is entitled to a large number of social security benefits, such as support for single-parent families. However, the field with the citizen's marital status was not really useful because it was not accompanied by the date from which the marital status was valid. (2) *Usability problems*. A facility or piece of information was present in the system but the user remained unaware of it, misinterpreted it, or had trouble using it. Example: When searching for a citizen's social security number on the basis of his or her name a checkbox enabled the user to indicate whether the full name was given or just part of it. The caption of this checkbox, ‘Search for patterns’, was unintelligible. (3) *Program bugs*. A facility or piece of information was present in the system but due to a program bug it did not behave as intended or did not work at all. Example: The same social security number could appear several times in the drop down list containing the citizens for whom information had most recently been displayed. (4) *Other*. This catch-all category contained only one problem which was related to the system configuration.

To indicate whether or not the problems were solved, each problem was assigned a status: (1) *Solved*, the problem was fixed. (2) *Reduced*, the problem was partly, but not fully, fixed. (3) *Unaddressed*, the problem was either deferred or rejected. A similar assessment of problem status was made during the project to maintain an overview of the progress made. The primary systems developer and either a usability specialist or the project manager made this assessment. The assessment made during the project and the subsequent coding made for this article assigned the same status to 88% of the problems, the remaining 12% were reassigned to indicate that they had been addressed to a lesser extent.

4.2 Activities

The activities of the primary systems developer were tracked in a diary (for a discussion of the diary method see Rieman, 1993). The diary, which was updated successively throughout the day, covered the nine-month period from the first user test through the fifth and contained

[1] Note that the definition of usability used in this study is narrower than the definition in ISO9241-11.

Table 1. The user tests

User test	Number of users	Offset from project start	Test duration	Test conducted by
Laboratory test	6	5 months	2 days	Usability specialists
Workshop test	8	8 months	1 day	Developers
Field test 1	8	10 months	3 weeks	Users
Field test 2	8	12 months	2 weeks	Users
Field test 3	8	13 months	5 weeks	Users

every activity with a duration of 15 minutes or more. The recordings were made on diary sheets, one for each day, and gave the starting and ending time of the activity, the project to which the activity pertained, and a terse description of the activity (see Figure 2 for an example of diary entries). To achieve this level of detail the current diary sheet was lying easily accessible on the developer's desk.

To enable investigations of when a problem was addressed and how long it took to correct it, each activity recorded in the diary was linked to the problems it addressed. Some activities were not performed in response to any of the problems identified by the users; other activities contributed to the solution of several problems. Thus, an activity could be linked to any number of problems, just as a problem could be linked to any number of activities.

5 The User Tests

The F&N project evolved around a series of user tests, each a major project milestone (see Table 1). Though different in many respects the five user tests shared the defining characteristic that a group of target users got hands-on experience with a running system prototype and expressed their opinion about it. The users participating in the tests were regular users of the mainframe version of the F&N system.

5.1 The Laboratory Test

The first user test was conducted by usability specialists in the in-house usability laboratory and involved six users who were asked to think out loud while solving eleven set tasks. Each task consisted of a brief description of a realistic scenario followed by a question, e.g. to find a specific piece of information. The test sessions, one for each user, consisted of a short introduction, 1½ hours of testing, and a debriefing interview. While working with the tasks the users were alone in the test room, and the two usability specialists conducting the test were in the control room which was separated from the test room by a one-way mirror. The users were frequently asked questions such as "What do you think that command will do?", "What would you expect to see or be able to do at this point?", and "What do you think the information on that part of the screen is telling you?" The usability specialists recorded problems observed during the sessions and communicated them to the development group in a test report. In addition, the primary systems developer observed the test sessions from an observation room. The sessions were also videotaped but the videotapes were not used during analysis, they were just back-up.

12:50 - 13:10	F&N	Write a help topic for [the online help writer].
13:10 - 14:00	F&N	Add the possibility of opening the window 'Information about authority' by double clicking a line in the list of back information about a citizen, including validation of any social security number in that line.
14:00 - 15:10	[Another project]	Assist [a colleague] in the proper use of [a reusable component developed as part of the F&N project].

Figure 2. An excerpt of the diary sheet for August 3, 1995

5.2 *The Workshop Test*

The second user test was also performed in-house but under the management of the development group and in a conference room rather than the usability laboratory. The workshop test began with a guided tour of the F&N system, performed by the project manager and the two systems developers. Then the users had two two-hour sessions for testing, separated by lunch. Finally, the test was concluded by a plenary discussion. Eight users participated in the test and they worked two by two on a number of set tasks, which collectively included five scenario descriptions each followed by a series of about ten specific questions. Each pair of users sat at a separate table with one computer and access to the printer within the room. When the users discovered a problem they either called upon a developer to report it directly or made a print-out of the screen and annotated it. The developers circled among the users to observe, inquire, and receive feedback. When the developers discovered a problem they approached the user for further details. After the test the development group produced an annotated list of the encountered problems.

5.3 *The Field Tests*

The third, fourth, and fifth user test were performed on site and managed by the users themselves. The users, the same eight persons as in the workshop test, had the F&N system installed at their personal workplace and used it occasionally in the execution of their day-to-day duties. They also had opportunity to discuss the system with their colleagues. There was no set tasks to be solved during these tests and the users did not keep a log of how much time they spent testing the system. The developers contacted the users once or twice during a test to ensure that everything was in working order, motivate further testing, and get feedback. Problems discovered by the users were reported by telephone or on the test form to be returned at the end of the test. The development group concluded each test by compiling an annotated list of the reported problems.

5.4 *The Rationale for the User Tests*

The laboratory test and the workshop test were carried out while the F&N system was under formation and receptive to suggestions for both minor and major modifications. The purpose of these two tests was quite similar. A major reason for choosing the workshop test the second time was that it was independent of the busy schedule of the laboratory. The remaining user tests were progressively more concerned with errors, at the expense of inconveniences. The first field test was performed to expose the F&N system to real-life conditions and, thus, have it evaluated in the context of the users' day-to-day duties, workload, and technical environment. The second field test was an informal acceptance test, and the third was the formal acceptance test intended to confirm that the system was ready for release.

6 Discussion

The diary contained 604 hours of work on the F&N project (51% of the working hours) distributed over 129 of the 164 working days covered by the diary. Most of the remaining time was spent on three other projects. The user tests encountered a total of 77 problems, distributed quite unevenly across the tests (see Table 2). One reason for the different number of problems found in each test was that the tests formed a sequence where one test followed another when all or most problems from the previous test had been addressed. Hence, it *cannot* be inferred that the laboratory test was superior to the workshop test, which in turn was superior to the field tests.

6.1 The Ecology of Set Tasks

The user tests uncovered a notably different mix of utility problems, usability problems, and program bugs (see Table 2). Of the problems uncovered by the laboratory test 76% concerned the usability of the system. Probably, a major reason for this was the use of set tasks, which were solved one by one without much digression. Set tasks tend to preclude discussion of whether the system lacks support for some aspects of actual tasks (Wright & Monk, 1991). Further, several users seemed to feel a remarkable pressure to perform during the laboratory test, even though they were told that the object of the test was the system, not the person using it. Under such circumstances the users cannot be expected to notice shortcomings of the set tasks or digress much from them to try other things. The users digressed from the ideal way of solving the tasks—they got into problems and recovered from these problems—but they kept pursuing the tasks and did so with little attention to their ecological validity. Hence, the laboratory test was biased toward usability at the expense of utility. Another consequence of the users' narrow focus on the set tasks was that few program bugs were encountered since the users stayed within the well-tested parts of the system.

The workshop test made use of set tasks too but only 20% of the encountered problems were usability problems. Instead, utility problems and program bugs each made up 40% of the problems. This seems to indicate that the users felt free to go beyond the set tasks and explore additional aspects of the system. In doing so they tested the system against their actual tasks, and they exercised the system in ways not foreseen by the developers. It seems reasonable to ascribe the users' more exploratory attitude to two circumstances:

- Working two by two the users were not alone when they got stuck or in doubt, and differences in their day-to-day work practices fostered discussion and divergent suggestions for solving the tasks.
- The informal atmosphere brought about by the presence of several other users, the face-to-face way of communicating with the persons conducting the test, the one-hour lunch break, and the absence of detailed observation of the users' behaviour.

The workshop test showed that the implications of set tasks were very much dependent upon the test situation in which the tasks appeared. The workshop test also showed that the developers were able to conduct a user test of their own system with results that made the test worth the effort. The workshop test was, however, restricted by leaving it almost entirely to the users to detect the problems, a restriction most obviously addressed by calling in a usability specialist skilled in observing users and spotting their problems.

Collectively the field tests uncovered a broad mix of problems but while the first and third field test were reasonably effective the second failed completely. Field test 3 uncovered a number of utility and usability problems and seemed successful in testing the system against the users' actual tasks. The test uncovered for example several problems relating to the print-

Table 2. Problem classification

Test	Utility problem	Usability problem	Program bug	Other	Total problems found
Laboratory test	8	29	1		38
Workshop test	8	4	8		20
Field test 1	1	2	5		8
Field test 2					0
Field test 3	4	6		1	11
Total	21	41	14	1	77

Note. Since the F&N system evolved from one test to the next the total number of problems found during a test is *not* evidence that one test is better than another.

outs produced by the system. Unintentionally, the set tasks used during the in-house user tests were so focused on the software that the users were not asked to evaluate the print-outs. This highlights a fundamental limitation of set tasks: They make tests blind to aspects not covered by the tasks. The low cost and valuable output of the field tests confer with the findings of Smilowitz et al. (1994), but the F&N project provides no support for their suggestion that the field test method may be improved further by providing set tasks.

6.2 Robustness and the Management of User Commitment

The effectiveness of user tests is critically dependent upon the active participation of the involved users. One way to control this dependency is to prescribe the users' behaviour in detail and carefully observe their execution of these prescriptions, i.e. the approach of the laboratory test. Leaving more decisions and initiative to the users, tests become increasingly dependent upon the individual user's personal motivation to do a good job. Field tests owe their low cost to leaving almost everything to the users, and precisely for that reason the essential task left with the persons conducting field tests is the management of the users' commitment to perform a thorough test. This is an indirect way to strive for robustness, but without supervision and a controlled environment it is practically the only one left.

The laboratory test and the workshop test had scheduled, supervised sessions dedicated to testing, and the detected problems were reported immediately. This way little effort was required on the part of the users to set up these tests and report their outcome. During the field tests it was the users' responsibility to devote some time to testing, and the procedure for reporting problems was more laborious in that it required either describing the problems in writing or phoning one of the developers. Field test 1 was the first time the F&N system was exposed to real-life conditions and it was accompanied by a strong commitment from the development group to fix a substantial fraction of the problems encountered. This context and one or two phone calls from the developers during the test motivated the users to spend some time testing the system. Field test 2 did not have a clearly stated purpose that differentiated it from the other tests. It merely asked the eight users from the two foregoing tests to test the system for the third time, and apparently that did not occasion the necessary enthusiasm. Users participating in a test have a long-term interest in the quality of the system since they will, probably, also be users of the released system, but they need further encouragement. It is the responsibility of the persons conducting a test to convince the users of its importance, otherwise busy users are unlikely to give a test priority at the expense of their day-to-day duties. Field test 3 was unique in that it was performed to decide whether the F&N system could be released or had to go through another iteration. The formal purpose of this test motivated the users since it was probably their last opportunity to affect—this release of—the F&N system.

Since the F&N project concerned the development of a new front end for an existing application the same data were available during the field tests as through the existing mainframe version. Data entered in either version could be viewed with the other. Thus, the users could shift freely between the old and the new version without loss of data or need of rekeying. This complete access to production data was, however, not exploited in the setup of the field tests though it seems that, for example, one-hour sessions where the new version was used in place of the old would have been a cogent tool in the management of the users' commitment. The complete access to years of production data distinguishes the F&N project from the development of new systems from scratch, but it should be noted that vast numbers of projects involve the development of new versions of existing systems. Zirkler & Ballman (1994), who consider field tests necessary to effective user testing, manage user commitment by physically arriving at the users' workplace and conduct the test. Though this is more costly than the field tests of the F&N project, Zirkler & Ballman report significant cost savings over the in-house usability tests they used to run.

6.3 Problem Impact

In the F&N project, 55 problems were solved or reduced while 22 problems were left unaddressed. To take the reduced problems into account the impact ratio calculation in Sawyer et al. (1996) was modified by counting the reduced problems as 50% solved:

$$\text{Impact ratio} = \frac{\text{Solved problems} + 0.5 * \text{Reduced problems}}{\text{Total problems found}} * 100$$

The overall impact ratio was 65%. This is comparable to the 78% reported by Sawyer et al. (1996) and to the 65% and 48% found by Whiteside et al. (1988) for their early in-house tests and subsequent field tests, respectively. However, the impact ratio varied considerably from one user test to another (see Table 3). Looking at the impact ratios of the individual user tests it is striking that only the three first tests had an impact while half of the unaddressed problems were uncovered during field test 3. This reflects that the field tests were increasingly concerned with critical problems only. Problems perceived to be non-critical were more and more often recorded and left for the next version. The 22 unaddressed problems form four groups: (1) Six problems were considered not to be predictive of actual use. Example: It was suggested to add a second confirmatory step after the user had confirmed a new note by pressing 'OK' rather than 'Cancel'. (2) Six problems stemmed from circumstances beyond the developers' influence. Example: The font size, prescribed by a mandatory corporate standard, was considered too small by several users. (3) Five problems were considered to be merely cosmetic. Example: The left margin of the print-outs was slightly narrow. (4) Five problems were left unaddressed because it was considered more important to get the front end released.

A very influential factor in determining whether or not a problem was addressed was *when* the problem was found: Finding a problem early profoundly increased its chances of being addressed. In the beginning and middle of a project much work remains and the project will be one of the project members' major concerns for some time to come. Also, many minor problems can be corrected at almost no extra cost when they can be addressed along with other problems concerning the same part of the design. Near the end of a project most project members spend the majority of their time on other projects or they are about to enter other projects, and little room and will is left for prolonging the old project even moderately. This means that relative to the project members' other responsibilities the time required to solve a user test problem tends to appear reasonable in the beginning or middle of a project and prohibitive near the end of the project (see also Kumar, 1990). Often, the last user test will have a 0% impact ratio because any modifications made trigger an extra test to assess the quality of these modifications.

Table 3. Problem status and impact ratio

Test	Solved	Reduced	Unaddressed	Total problems found	Impact ratio
Laboratory test	25	6	7	38	74%
Workshop test	13	3	4	20	73%
Field test 1	7	1		8	94%
Field test 2				0	-
Field test 3			11	11	0%
Total	45	10	22	77	65%

In the F&N project the single-most important factor in ensuring a high impact of a user test seems to be to avoid performing the test during the last third of the project. The low-cost, unsupervised field tests performed in the F&N project can only be conducted late in the project when the system prototype is fairly stable. This suggests that the price of the low cost of these tests is that they will usually have a low impact. A potentially attractive alternative is supervised field tests where a developer or usability specialist accompanies the system in the field to handle problems with the system and to observe. Supervised field tests can be performed earlier and allow for better management of the users' commitment, but they require more resources and the presence of an evaluator introduces an ecological gap.

6.4 Resources Spent Finding Versus Addressing Problems

The total cost of user testing is the cost of conducting the tests plus the cost of addressing the detected problems. In the F&N project, the laboratory test was the more resource-demanding test in terms of equipment, expertise as well as person-hours, and the field tests required the fewer resources. More notably, the cost of addressing the problems was quite substantial. The primary systems developer spent 25% of his time fixing problems encountered during the five user tests (see Table 4). At the time of the laboratory test several facilities were not yet developed. Thus, the problems found during this test were added to an already long list of outstanding tasks. As the project progressed the list got shorter and increasingly dominated by the input from the user tests. To the primary systems developer this meant that user test issues came to occupy more of his time. Near the end of the project the action taken on the user tests was restricted to the presumably critical problems and the amount of time spent on user test issues dropped. Averaged over the entire project the primary systems developer spent 2 to 3 hours a problem, but the time spent on the individual problems varied a lot. For field test 1 the average time spent to detect a problem was half an hour. Thus the cost of addressing the problems encountered during this test clearly exceeded that of finding them. Similar figures cannot be given for the other tests because field test 1 was the only test administered by the diary-keeping, primary systems developer alone. However, the average time spent to detect a problem was most likely higher for the laboratory test and the workshop test.

During the five months from the laboratory test to field test 1 the primary systems developer spent an average of more than five hours a working day on the F&N project. During the remaining four months of the project it occupied substantially fewer daily hours. A major reason for this decrease in project intensity was the duration of the field tests. To allow the users time to fit the field tests into their schedule and get to actually use the system for some time, the field tests lasted 3, 2, and 5 weeks. During these periods the list of outstanding tasks contained few, if any, high-priority tasks and little work was done on the F&N project. These periods of waiting allowed the project members to devote their attention to other projects, but these periods were also a significant cost of the field tests because they prolonged the F&N

Table 4. Time spent by the primary systems developer, in total and to address the problems encountered during the user tests

Period	Total hours spent	Total hours spent a working day	Hours spent fixing problems	Percent of time spent fixing problems
Laboratory test - workshop test	279	6:29	56	20%
Workshop test - field test 1	216	4:48	70	32%
Field test 1 - field test 2	79	2:09	25	32%
Field test 2 - field test 3	4	0:40	0	0%
Field test 3 - system released	26	0:47	1	4%
Total	604	3:41	152	25%

project. The way to reduce this cost is to make the field tests more efficient, i.e. to obtain the same benefits in a shorter span of time. This is to a certain extent a trade-off between spending resources passively by prolonging the projects and spending resources actively, for example by conducting supervised field tests in order to manage the users' commitment better.

7 Conclusion

This study has investigated the effectiveness of the five user tests conducted during the development of a graphical front end for the F&N system. Overall the user tests were effective and led to numerous improvements of the front end, but the tests differed substantially in terms of how they balanced resources, impact, robustness, and ecological validity. This study concerns the F&N project and no strong claims can be made as to the generality of the findings. The F&N project employed an iterative, user-centred approach and concerned the development of a new version of an existing application. It is reasonable to assume that the findings can be made subject to some generalisation if a development process of a similar nature is studied.

Tests like the laboratory test are designed to control variability and thereby achieve robustness. This is costly in terms of resources such as equipment, expertise, and person-hours, and it introduces a number of threats to the ecological validity of the test. The laboratory test relied on the use of set tasks to direct the users' activities and provide the evaluator with knowledge about what the users were trying to achieve. However, the formality of the laboratory test seemed to place the users under a pressure that precluded considerations about whether the tasks were representative of the users' actual work. That is, the test sessions provided little basis for evaluating the ecological validity of the set tasks. This made the laboratory test less suited early in the development process where the utility of the system, i.e. *what* the system can do, was the major design concern. The laboratory test was more concerned with usability, i.e. *how* tasks are performed with the system.

The workshop test focused more on utility, and it seems probable that this focus was nourished by the more informal test situation with plenty of room for discussion. For that reason it should be considered to swap the workshop test and the laboratory test. It is however worth noting that much of the formality of the laboratory test could be removed at no cost, for example the evaluator could have been in the test room with the user since they talked much anyway. The workshop test combined set tasks and direct supervision with an informal atmosphere and co-operating users. This way the test unfolded around the set tasks with frequent exploration of issues that went beyond the tasks. Since the workshop test left the detection of problems almost entirely to the users, the test did however miss the problems where the users themselves were not aware of their difficulties or ascribed them to their lack of experience in using the system.

The field tests displayed dramatic differences in their contributions to the project. Larger robustness is utterly needed, and since field tests are characterised by leaving almost everything to the users the one, essential task left with the persons conducting unsupervised field tests is the careful arousal and management of the users' commitment to perform a thorough test. Few development tasks could be done in parallel with the field tests since the system had to be rather complete before it could be tested in the field. For that reason, the field tests prolonged the project by introducing week-long periods where the project merely awaited the results of the field tests. These periods of waiting added a substantial cost to the field tests which otherwise required very few resources.

The user tests that were conducted early in the development process had a much higher impact than those conducted near the end of the project where the project members had got seriously involved in other projects. This affected especially the field tests, and it highlights the importance of designing user testing methods in ways that allow them to be applied early. Altogether the user tests had a substantial impact on the focus of the entire development effort in that 25% of the primary systems developer's time was spent solving problems encountered during the tests. This gives an indication of the cost of addressing the encountered problems, and it suggests that the amount of user testing that can be done in a project is limited by the cost of addressing the problems, rather than by the cost of conducting the tests.

8 Acknowledgements

This work has been supported by a grant from the Danish National Research Foundation. I wish to thank Niels Jacobsen and Erik Frøkjær for their valuable comments to earlier drafts of this article. Special thanks are due to the members of the F&N project group, the usability specialists, and the users who participated in the user tests.

9 References

Bailey, R. W., Allan, R. W., & Raiello, P. (1992). Usability testing vs. heuristic evaluation: A head-to-head comparison. In *Proceedings of the Human Factors Society 36th Annual Meeting* (pp. 409-413). Santa Monica: Human Factors Society.

Bias, R. G., & Mayhew, D. J. (Eds.). (1994). *Cost-justifying usability*. Boston: Academic Press.

Brooks, P. (1994). Adding value to usability testing. In J. Nielsen & R. L. Mack (Eds.), *Usability Inspection Methods* (pp. 255-271). New York: John Wiley.

Hackman, G. S., & Biers, D. W. (1992). Team usability testing: Are two heads better than one? In *Proceedings of the Human Factors Society 36th Annual Meeting* (pp. 1205-1209). Santa Monica: Human Factors Society.

Held, J. E., & Biers, D. W. (1992). Software usability testing: Do evaluator intervention and task structure make any difference? In *Proceedings of the Human Factors Society 36th Annual Meeting* (pp. 1215-1219). Santa Monica: Human Factors Society.

Holleran, P. A. (1991). A methodological note on pitfalls in usability testing. *Behaviour & Information Technology*, 10, 345-357.

Jacobsen, N. E., Hertzum, M., & John, B. E. (1998). The evaluator effect in usability studies: Problem detection and severity judgments. In *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting* (pp. 1336-1340). Santa Monica: Human Factors and Ergonomics Society.

Kumar, K. (1990). Post implementation evaluation of computer-based information systems: Current practices. *Communications of the ACM*, 33(2), 203-212.

Lewis, J. R. (1994). Sample sizes for usability studies: Additional considerations. *Human Factors*, 36, 368-378.

Lund, A. M. (1997). Another approach to justifying the cost of usability. *ACM Interactions*, 4(3), 48-56.

Nielsen, J. (1993). *Usability engineering*. Boston: Academic Press.

O'Malley, C. E., Draper, S. W., & Riley, M. S. (1984). Constructive interaction: A method for studying human-computer-human interaction. In *Proceedings of the IFIP INTERACT'84*

First International Conference on Human-Computer Interaction (pp. 269-274). Amsterdam: Elsevier.

Pejtersen, A. M., & Rasmussen, J. (1997). Effectiveness testing of complex systems. In G. Salvendy (Ed.), *Handbook of Human Factors and Ergonomics* (pp. 1514-1542). New York: John Wiley.

Rieman, J. (1993). The diary study: A workplace-oriented research tool to guide laboratory efforts. In *Proceedings of the ACM/IFIP INTERCHI'93 Conference on Human Factors in Computing Systems* (pp. 321-326). New York: ACM Press.

Rowley, D. E. (1994). Usability testing in the field: Bringing the laboratory to the user. In *Proceedings of the ACM CHI'94 Conference on Human Factors in Computing Systems* (pp. 252-257). New York: ACM Press.

Rubin, J. (1994). *Handbook of usability testing: How to plan, design, and conduct effective tests*. New York: John Wiley.

Sawyer, P., Flanders, A., & Wixon, D. (1996). Making a difference - The impact of inspections. In *Proceedings of the ACM CHI'96 Conference on Human Factors in Computing Systems* (pp. 376-382). New York: ACM Press.

Schrier, J. R. (1992). Reducing stress associated with participating in a usability test. In *Proceedings of the Human Factors Society 36th Annual Meeting* (pp. 1210-1214). Santa Monica: Human Factors Society.

Smilowitz, E. D., Darnell, M. J., & Benson, A. E. (1994). Are we overlooking some usability testing methods? A comparison of lab, beta, and forum tests. *Behaviour & Information Technology*, 13, 183-190.

Szczur, M. (1994). Usability testing—on a budget: A NASA usability test case study. Behaviour & Information Technology, 13, 106-118.

Thomas, J. C., & Kellogg, W. A. (1989). Minimizing ecological gaps in interface design. *IEEE Software*, 6(1), 78-86.

Whiteside, J., Bennett, J., & Holtzblatt, K. (1988). Usability engineering: Our experience and evolution. In M. Helander (Ed.), *Handbook of Human-Computer Interaction* (pp. 791-817). Amsterdam: Elsevier.

Wright, P. C., & Monk, A. F. (1991). A cost-effective evaluation method for use by designers. *International Journal of Man-Machine Studies*, 35, 891-912.

Zirkler, D., & Ballman, D. R. (1994). Usability testing in a competitive market: Lessons learned. *Behaviour & Information Technology*, 13, 191-197.

Universal Access in the work-place: A case study

Simeon Keates[1], John Clarkson[1], Joanne Coy[2] and Peter Robinson[3]

[1] Department of Engineering, University of Cambridge
[2] Post Office Research Group, The Post Office
[3] Computer Laboratory, University of Cambridge

Abstract

Universal Access belongs not only in the research laboratory, but also in an industrial environment. Many countries have set out specific legal requirements for companies to meet in terms of proportions of staff with disabilities and meeting those requirements is a new experience for many employers. More enlightened companies wish to take their commitment beyond meeting the letter of the law and to provide a genuinely inclusive work-place. However, implementing such an approach is complex.

The principal aim of this paper is to describe the steps being taken by The Post Office (TPO), with the assistance of the University of Cambridge, to offer a more inclusive work-place to support a wider range of employee physical capabilities. Computer access, including interface and input system design issues, features very prominently in research at Cambridge. An approach taken to assess the potential for universal access within the TPO environment is described and the relative merits of adaptive and proactive design methods discussed. Although the case study presented is very specific and still in its preliminary stages, the generic usability issues are applicable to a wider range of computer applications.

INTRODUCTION

In the United Kingdom, all companies with over 25 employees are expected to meet the requirements of the 1996 Disability Discrimination Act [DDA 96]. This stipulates a minimum proportion of employees with physical impairments and that the work-place must not present unnecessary obstacles to employment opportunities. This includes both buildings access and access to work opportunities. It is the latter target, access to employment opportunities through the appropriate provision of universally accessible equipment, that this paper focuses on. by including the specification of equipment used. Similar acts exists in many other countries, most noticeably the 1990 Americans with Disabilities Act [ADA 90].

However, providing an inclusive work-place is not simply a matter of meeting legislated requirements. There are strong economic arguments why employers should wish to remove unnecessary barriers to employment. For example, the ageing population is growing inexorably. By 2020, almost half the adult population in the UK will be over 50, with the over 80's being the most rapidly growing sector [Coleman 93] and with age comes an increasing divergence of physical capability [Fozard 90]. It will become increasingly important for employers to ensure that they do not arbitrarily curtail an employees working life simply because of an inaccessible work-place.

Technological solutions to inclusive work-places

Computers continue to play an increasing role in the typical office and ensuring *universal access* to them is a key goal. If designed correctly, they have the potential to enable employers to retain the services of experienced employees who might otherwise have to retire through ill health. In addition they may enable recruitment of those with physical impairments.

Other approaches to making the work-place more inclusive include the development of assistive robotic devices [Buhler 98]. If designed appropriately, they can help provide vocation opportunities to those with the mental faculty to perform useful tasks, but whose physical impairments would otherwise prohibit their employment. Examples include the AFMASTER office workstation [Gelin 99] and the IRVIS robot [Dowland 98].

A more fundamental change in philosophy is to encourage working at home. The technology required to support this is now readily available and removes the need for operators with limited mobility to commute on a daily basis. The principal advantage of working at home is that it should enable employees to work in an environment that has been specifically tailored to meet *their* individual needs. Perhaps it could be thought of as the ultimate inclusive work-place.

Other perceived advantages of working at home include offering a degree of insulation from work-place stresses. For example, it avoids negative reinforcement of the motion-impaired operator's abilities when sitting next to an able-bodied operator who is possibly working at a higher rate of productivity.

However, there are also disadvantages to working at home, particularly a sense of isolation and a lack of communication with colleagues. Unless specific support measures are included to assist with a sense of being part of a team, then motivational issues may become preeminent as there is a risk that the operator may feel left out. This may be overcome by using the same technology that enables the remote access to work to provide access to social structures, such as virtual communities [Pieper99].

UNIVERSAL ACCESS AND THE POST OFFICE

The Post Office (TPO) is one of the UK's largest employers, directly employing some 190,000 people. It consists of four core businesses - Royal Mail, Parcelforce Worldwide, Post Office Counters and Subscription Services Limited. Of particular interest in this paper is the Royal Mail, which is responsible for the delivery of the overwhelming majority of letters and small parcels within the UK. The usual sequence of delivery involves the posting of a letter at a postbox or over the counter in a post office, collection by van and transportation to a centralised sorting office for automatic sorting. From there the letters are despatched to the appropriate local delivery office for distribution by postmen and women.

As with all large UK companies, TPO is required under the 1996 DDA to have at least 3% of its workforce classified as disabled. This equates to approximately 6000 employees for whom gainful employment opportunities must be found. In addition, Royal Mail employees exhibit the usual prevalence of conditions and disabilities associated with ageing, such as strokes and other neurological conditions, but with a particular tendency towards developing arthritis. With a community of employees that is largely involved with travelling, there is also a higher than average rate of trauma through accidents. TPO currently has a medical retirement rate that costs the company over £100,000,000 last year. There is clearly a strong incentive to retain existing staff for as long as possible, both to reduce this cost and to extend the productive work-life of their employees.

The Royal Mail activities are predominantly physical in nature, from the letter collection to delivery. As such, they are not easily accessible by those with motion impairments or suitable for adaptation. Consequently, the TPO's approach has focused on areas providing support for the collection and delivery processes. Computers form a major part of such systems and computer support is seen as the means to creating a more inclusive working environment. TPO have already invested in the development of an in-house Disability Advice Centre, which recommends hardware accessibility options to those unable to use the traditional keyboard and mouse input arrangement.

The development of software interfaces and input systems that offer universal access is clearly an essential component in facilitating an inclusive work-place and the following case study highlights the many of the issues facing employers intent on creating such conditions.

CASE STUDY: UNIVERSAL ACCESS AND MAIL CODING

Video mail coding is an activity that supports the delivery process of the Royal Mail. It can be performed either on-line or off-line and is well suited to modification for universal access.

The need for video mail-coding

During automatic sorting for delivery, letters are fed past four digital cameras that take two images of each side of the envelope. These images are processed by an optical character recognition (OCR) suite that attempts to locate and read the postcode. If the OCR software fails to recognise the postcode to a predetermined level of confidence, then the images are fed to a Mail Coding Suite, where an operator manually enters the postcode. The typical postcode format is *letter-letter-number number-letter-letter*, where the numbers can have one or two digits. An example would be CB12 5ZF, where the area is defined by the CB12 and the street location by the 5ZF.

A 14 second delay is built into the letter sorting machine specifically to allow time for the images to be fed to the Mail Coding Suite and the postcode to be entered. After this, the mail is turned the right way up by recognising the position of the stamp and tagged either with a short four state bar code if the postcode is recognised, or a longer tag code if not. The letters are then sent onward for distribution or, if the postcodes are not identified during the 14 second delay, gathered and removed from the sorting machine. They are then stored until the postcodes have been processed. Between 20% and 70% of letters are successfully coded by the OCR system, the precise rate being affected by large business distributions and seasonal factors, such as Christmas.

The current mail coding process

The coders sit in a dedicated office, facing a terminal with a portrait orientated monochrome monitor. The keyboard is a standard layout, but the key tops have been customised for the coding process. Operators generally work 50 minutes in every hour although not for extensive periods of time. Their duties are intermixed with overseeing the activities of the sorting machines.

The monitor display is divided into three sections. The lower section displays the digitally captured images of the envelope. The centre strip displays the text entered by the coder along with prompts for action, and the top shows pop-up menus as required. The image of the envelope is clear and easy on the eye. There is no attempt to assist the coders in locating the postcode. This has the advantage of not misleading the coder if the OCR has failed to locate the code correctly, but has the disadvantage that it would be cognitively faster for the coder to focus on a highlighted object.

The actions required from the coder are the same irrespective of whether the OCR has recognised most of the characters, or conversely failed to even locate the code. This has the advantage of minimising the cognitive effort required by the coder. However, it does lead to unnecessary key inputs if the OCR is only slightly wrong.

The maximum time that an operator has to enter the postcode on-line is theoretically 14 seconds from the scanning to the tagging of each letter in the sorting machine. Each postcode can be up to 8 characters long, giving 1.75 seconds per keystroke. If this time limit is not met then the letter has to be taken off-line to be coded later. On a busy day, with increased throughput and multiple letters that need to be coded entering the delay run, this time can be decreased to approximately 3 to 4 seconds per postcode. This corresponds to 500ms per keystroke.

Mail coding and the motion-impaired user

Mail-coding is inherently a process which a motion-impaired user should be able to do since the task is principally visual. In addition, the data is available in an electronic format and, with remote access technology, the user does not have to be located geographically close to the sorting machine. The volume of data to be transmitted, though, would be more suited to off-line coding.

For motion-impaired users to be able to contribute to the coding process, it is necessary to address the issue of the data entry rates for on-line coding. It is known that a typical response for a motion-impaired user to perform a keystroke in response to a simple stimulus is approximately 620ms [Keates 98]. This time does not allow for deciding which key to press or locating it on the keyboard. It is therefore unlikely that a motion-impaired user could match the 500ms data entry rates required to perform the coding on-line and certainly not for any extended period of time.

More mildly impaired users may be more able to meet these times, but this would have to be determined on a case-by-case basis. Reductions in the available time for postcode entry through increased letter-flow or a burst of poorly packaged circulars, would invariably mean that the gap between desired and actual productivity rates would widen. Even with radically different input hardware, it is unlikely that the necessary time constraints could be met whilst retaining the current data entry requirements.

Adapting the existing coding system

Discussions have shown that traditional HCI approaches are not the best way to achieving a universal access and that more specific studies are required to reach this goal [Stephanidis 97]. Two core themes from this work are *proactivity* - addressing the issue of accessibility at design time, and *adaptation* - the ability for the interface to be tailored to the user [Stary 97].

For the mail coding, as with all universal access software development, a choice has to be made between proactive and adaptive design measures to ensure accessibility. Possible adaptive input system modifications might include:

- Voice input – this removes a substantial portion of the impairment-related difficulties, but is unlikely to be effective. Even when ignoring the issues of motion-impaired operators who also have an associated voice impairment (not generally the case for arthritis, but certainly for cerebral palsy, post-stroke, and so on), this is a task that does not involve standard spoken English, but clusters of letters and numbers. Consequently, an unnatural, rather stilted way of speaking would be required that the users may tire of quickly.
- Specialist keyboards – the existing keyboard has been modified for this application and the modifications could be kept. However, the physical layout of the keys, spacing, size, could be modified for each user. Keyguards are an option, as are small keyboards.
- Keyboard emulators – for those who simply cannot use a keyboard, an on-screen keyboard emulator is the obvious choice, used in conjunction with an appropriate cursor device, or in scanning mode with a switch.

Alternative approaches for assisting motion-impaired users

Taking a more proactive approach to interface development, new coding systems could be developed with appropriate software interfaces and input systems to allow motion-impaired users to contribute productively to the video mail coding activities. If the premise that on-line coding of full postcodes by motion-impaired users will not meet the stipulated time constraints is accepted, then methods need to be found to either relax the time constraint or improve the effectiveness of the operators within the allowed time frame. If either, or both, of these goals could be achieved, then this should enable motion-impaired users to participate more actively in the coding process. In order to achieve this, it is necessary for the interface to be more active in assisting the coder.

Relaxing the time constraint - extending the physical length of the delay run in the sorting machine would proportionally increase the time allowed for data entry, but is not economically viable. A corollary to this, to help exploit the full availability of the time delay, would be to have more people working on the coding, which would decrease the number of letters per operator that required video coding. The final variant would be to dynamically allocate letters just entering the delay, rather already in it, to the motion-impaired operators to give them the full 14 seconds available.

Alternatively, off-line coding provides an opportunity to increase the time available for coding. A final, and obvious, option is to increase the OCR recognition rate and reduce the percentage of letters needing video coding. This has the equally obvious disadvantage of being more easily stated than implemented.

Improving the operator effectiveness - the existing able-bodied coders rely principally on their speed of data entry to perform the coding within the available time. Cognitive processing of the information presented to them is kept to an absolute minimum in an attempt to make the data entry as automatic as possible. In effect, they try reduce the task to one of muscle-memory and so any action by the interface that distracts from this basic goal may be viewed as an interference.

However, reducing the amount of data that needs to be entered offers great potential for assisting motion-impaired users. Unlike the able-bodied coders, the time per physical keystroke for motion-impaired users is much higher than the cognitive time required to decide the input required. Consequently, minimising the amount of text entry for the motion-impaired coders is an essential goal. Methods for achieving this might include:

- Suggesting a code for the user to accept or decline when the OCR system is unsure of its interpretation of the postcode. This would be an interference for an able-bodied user, but not for a motion-impaired one. The success of this strategy would depend on the OCR system identifying the correct location for postcode and the threshold values for deciding whether to accept its own results.
- Presenting the user with a ‘show me where the postcode is’ request when the OCR system is confused about the location of the postcode. The operator could then highlight the postcode region on the envelope by a pointing device such as a mouse. This would be useful in the cases of a postcard or where there are multiple addresses on the envelope.
- Prompting the user with a ‘tell me the missing letters/numbers only’ request when the OCR has recognised some of the characters. Again this would probably be an interference for able-bodied users, but not for motion-impaired ones.

Other methods of improving the effectiveness of the motion-impaired coders could focus on increasing the speed of data entry. New input media, such as those described in the section on adaptive measures could be investigated.

Further work

To develop the interface beyond its current passive state, it is necessary to measure certain performance parameters for different user groups. Proposed areas of study include:

- an investigation of comparative rates of keystrokes for users with arthritis, neurological and congenital conditions;
- the quantification of the effects of re-distributioning the data entry from the high motor load/low cognitive effort of typing the whole postcode to a more active, low motor load/higher cognitive effort interface;
- the design and evaluation of new interfaces and input media;
- the development of a more complete model of human-computer interaction for users with disabilities.

CONCLUSIONS

Enabling inclusive work-places is not straightforward. Although legislation requires large corporations to employ people with physical impairments, only limited guidance is provided in how these targets should be met. Some organisations, such as The Post Office, view it not only as their legal, but also their ethical responsibility to meet the spirit, and not just the letter, of the law. However, until universal access becomes a de facto standard, to do so will require substantial investment in time and resources.

The case study described in this paper shows that retrospective adaptation of an interface is unlikely to be sufficient to provide universal access to the mail coding task and that a more comprehensive re-design of the entire interface is required. The challenge for research establishments must surely be to find methods to make adopting universal access as effective and economically viable as possible and to communicate this knowledge to industry.

ACKNOWLEDGEMENTS

The authors would like to thank the EPSRC (grant GR/M45177) and The Post Office for funding this research.

REFERENCES

[ADA 90] *Americans with Disabilities Act of 1990*, US Public Law 101-336, 1990.

[Buhler 98] C. Buhler, Robotics for Rehabilitation - A European(?) Perspective. *Robotica*, 1998 **16**(5), pp 487-490.

[Coleman 93] R. Coleman, A demographic overview of the ageing of First World populations. *Applied Ergonomics*, 1993, **24**(1), pp 5-8.

[DDA 96] *The Disability Discrimination Act*, Dept. for Education and Employment, 1996.

[Dowland 98] R. Dowland, P.J. Clarkson, R. Cipolla, A prototyping strategy for use in interactive robotic system development. *Robotica*, 1998, **16**(5), pp 517-521.

[Fozard 90] J. Fozard, E. Metter, L. Brant, Next steps in describing ageing and disease in longitudinal studies. *Journal of Gerontology*, 1990, **45**(4).

[Gelin 99] R. Gelin, F. Coulon-Lauture, B. Lesigne, J-M le Blanc, M Busnel, AFMASTER: An industrial rehabilitation workstation. *Proceedings of ICORR 99*, Stanford University, USA, 1999, pp 149-155.

[Keates 98] S. Keates, P.J. Clarkson, P. Robinson, Developing a methodology for the design of accessible interfaces. *Proceedings of the 4th ERCIM Workshop*, Stockholm, Sweden, 1998.

[Pieper 99] M. Pieper, Information environments to overcome isolation, marginalization and stigmatization in an overaging information society. *Proceedings of HCI International '99*, Munich, Germany, 1999, pp 883-887.

[Stary 97] C. Stary, The role of design and evaluation principles for user interfaces for all. *Proceedings of HCI International '97*, San Francisco, USA, 1997, pp 477-480.

[Stephanidis 97] C. Stephanidis, Towards the next generation of UIST: Developing for all users. *Proceedings of HCI International '97*, San Francisco, USA, 1997, pp 473-476.

Rapid Development and Evaluation of Interactive Systems

Jörg Marrenbach
Department of Technical Computer Science
Aachen University of Technology, Germany
Phone: +49-241-44026105 Fax: +49-241-8888308
Ahornstrasse 55, D-52074 Aachen
E-Mail: marrenbach@techinfo.rwth-aachen.de

Abstract – The interaction between humans and machines is ubiquitous in daily life. Some interaction techniques are more efficient or effective than others for certain tasks or users. During the development of new systems, the user is referred frequently too late into the development process. This is due to the fact that a prototype does not exist to verify the system by the user.

This article outlines some ideas regarding the relationship between the systems specification and the user requirements at this early stage of the design process through the use of an appropriate specification tool. This tool has been developed by the Department of Technical Computer Science that enables the designer to create models of tasks, users, and devices and to connect these models to visualise the relations between them.

1 Introduction

The interaction between humans and machines is ubiquitous in daily life. Some interaction techniques are more efficient or effective than others for certain tasks or users.

The traditional methodology within the development of software-based products was essentially based on a linear, phase-oriented life cycle. This approach is also called "waterfall" model. It consists of different levels for definition of requirements, specification, design, implementation, validation, verification, operation and maintenance. Such a model presupposes that customers or end users are able to formulate their conceptions very precisely at the beginning of the development process. This is not the case according to experience. The requirements are reflected to shorten the development process, to overlap the individual phases and enable continuous modifications.

This led from the linear to the iterative model of software development, in which in early phases creating of prototypes is possible already. ISO 13407 [1] describes such an user and task oriented development cycle (Fig. 1).

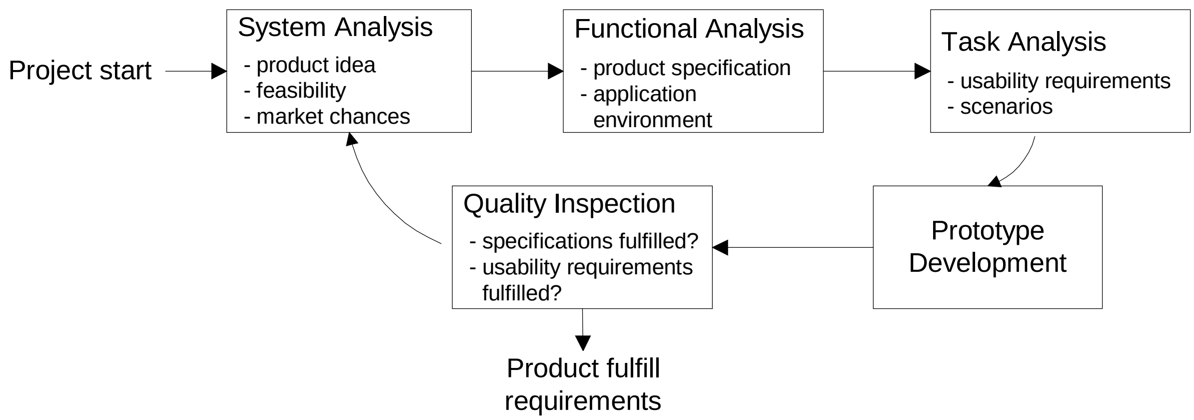


Fig. 1: Development cycle for interactive systems

First of all, the user requirements, their functions and the intended application environment are to be determined. From this analysis the usability requirements can be derived and from these results first product designs (prototypes) can be developed in a third step. These prototypes have to be verified for agreement with the specified usability requirements. Also end users must be queried. After the identification of further development needs the cycle starts again, until all requirements are fulfilled.

2 Motivation for a new Methodology

As system complexity increases, it becomes more difficult to completely specify detailed requirements in text form. The documents that attempt to describe these systems become large and complex. The requirements may interact in intricate and complex ways. The review and sign-off processes can be lengthy and expensive. Verifying that the requirements documentation is complete, accurate and consistent can be a daunting or even impossible task.

As the problems to be addressed increase the complexity, the solution approaches become less obvious. Software developers will not be experts in the domain of the problems to be solved, so it is similarly unrealistic to depend solely on them to define a system. A co-operative effort, among domain experts and technology experts, to discover system requirements can leverage the value added of new systems. Ultimately, the need for a better way to develop software systems is driven by the need to manage the risks involved, i.e., development costs.

3 Concept for Modern Software Development Environment

While many system development efforts still claim to use the waterfall model, in the trenches programmers, analysts and project managers are devising more effective techniques.

This section outlines a concept regarding the relationship between the system specifications and the user requirements at an early stage of the design process. The concept can make effective use of tools to integrate the requirements analysis, design, code and test environments. This concept enables the developer to create models of tasks, users and devices and to connect these models to visualise the relations between them. Based on these models an analytical evaluation of the system can be done. Additionally, high quality systems can be developed much faster using this concept as opposed to the waterfall approach, especially by using the different models. Furthermore, it can be assumed, that user satisfaction with the systems developed improves when using this concept.

While there is a wide range of tools for specifying device models and task models for more or less complex devices, no tool seems to offer an easy way to create an accompanied user model to a device prototype. Therefore in most cases, the first stage at which the designer can determine the quality of human machine interaction is when the device is fully built and real user studies are conducted. However, it would be much more convenient to build a normative user model as part of the device specification, and having a tool to evaluate the prototype.

4 System Architecture

For the evaluation of man-machine interfaces regarding the adequacy three criteria are considered as essential. According to ISO 9241 part 11 [2] these are named effectiveness, efficiency and satisfaction. These criteria determine the general evaluation frameworks for the investigation of man-machine systems. In addition, further specification demands are considered, as specified, e.g., in ISO 9241 part 10 [3].

With consideration of the above-mentioned criteria an environment is developed, which enables the developer to evaluate a technical system promptly regarding to ergonomic requirements. Figure 2 shows the dependencies of the development environment. On the basis of a functionality already specified in the requirements a device model is created. Subsequently a user model as well as a user interface are built up.

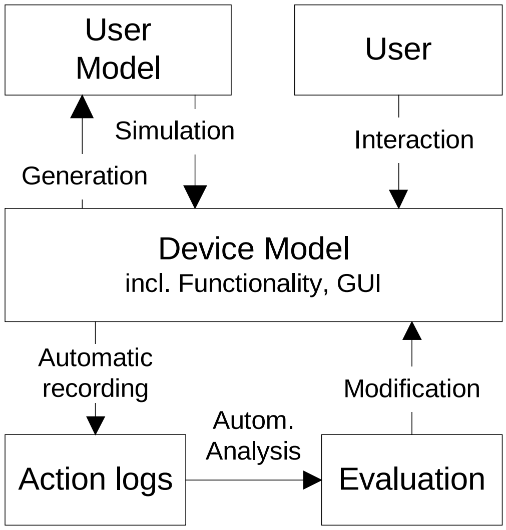


Fig 2: System architecture of the development environment

The evaluation can be done by a simulated use via the user model and by test with prospective users. Action logs are generated automatically and conclusions concerning the operability of the prototype can be drawn, e.g., from errors and learning times.

4.1 Device Specification

A usual device prototype consists of several connected and dependent models. The device model contains details about the inner works of the device, specified as states and transitions. For the specification of the device model statecharts are used [4]. Statecharts are an extension of usual transition diagrams, offering the possibility for hierarchical and concurrent state organisation [5]. A device is modelled as a set of states interconnected by action transitions. A task model describes what a user is expected to do with the device. This model is created by the systems designer and can be seen as an early form of the owners manual of the device [6].

4.2 User Modelling

The user model, while similar to the task model in its relation to user actions, is extended by specific values for the user workload and complexity of tasks. User models can be used for different purposes. While they are no integral part of a device, user models can help making devices easier to use. Special adaptive user models can even support the user actively in interacting with the device. For designing and prototyping matters, normative user models are of interest. They describe a normal user with average knowledge about the devices and its purpose. Contained in the normative model is a hierarchy of operations to be performed in order to achieve a goal [7].

5 StateWatch Approach

The tool currently under development, called *StateWatch*, is used to specify the device behaviour in form of graphical data diagrams (statecharts), to automatically generate source code from these diagrams and to run in real-time. Additionally, a graphical user interface (GUI) is created from the device model's states and actions, and the system designer is able to define a user model and add hints for task complexity and performance measurement. The created software prototype is executed directly from the tool, without writing or generating code manually, to see if it fits the users requirements. Action protocols are logged during test sessions automatically. The transitions of states is displayed graphically to allow the designer a better validation of the device model.

Furthermore, the normative user model created as part of the software prototype is able to rule the user interface elements, causing actions a test user is expected to perform. Protocols logged during these automated test sessions can afterwards be compared to real user logs, allowing for an easier evaluation.

5.1 Used Tools

Statemate is the application used to create statecharts for transition diagrams that specify the device behaviour [8]. Statemate is able to build up executable software versions of the modelled devices, however a real GUI creator is missing.

The market offers various products for designing user interfaces, but only some are able to simulate the created user interface directly without writing, generating, or compiling code. The GUI tool that is used as a basis for user model integration is called *InterfaceBuilder* [9]. StateWatch can be implemented as a palette extension to InterfaceBuilder. Several new interaction and control elements are located on the palette and can be integrated into a device prototype.

5.2 Technical Aspects

InterfaceBuilder is a tool that works in a object oriented way. The developer creates an application by dragging objects from palettes, arranging them on a workspace, and wiring them up graphically. StateWatch adds objects to this environment. There are common GUI objects for control lights, windows, buttons, etc. with somewhat changed behaviour compared to the default elements. In addition, objects for connection to Statemate (ActionWrapper) and for creating and editing a user model are available. Figure 3 depicts StateWatch as a palette extension to InterfaceBuilder.

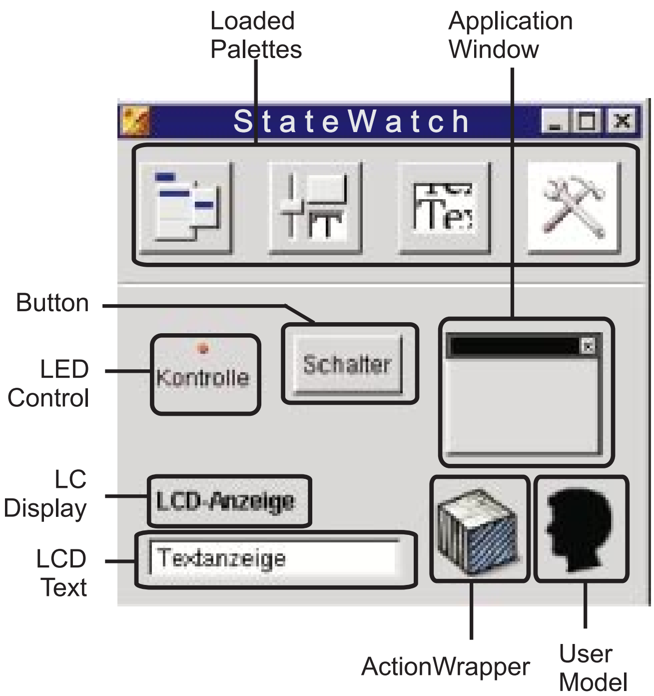


Fig. 3: StateWatch integrated into InterfaceBuilder

StateWatch requires a device model to be created with Statemate, specifying the device behaviour and possible user actions. The system designer can then create a GUI using the full InterfaceBuilder functionality, connecting the interface elements back to actions from the statechart. Output elements, called data sinks, can be connected graphically to data sources like other GUI elements or states.
The task analysis following the GOMS approach, a formal description technique [10, 11], is used for building the user model. StateWatch enhances the GOMS modelling technique in that way that complexity information can be added to operators. In that way the total complexity to achieve a goal can be determined during a test run the user model performs.

5.2 Application

A CD player prototype was created to test the StateWatch features. In general, all CD players are quite similar in use and it is save that to assume that the most people are familiar with their usage. Figure 4 shows the devices statechart modelled with Statemate and the related graphical user interface of the prototype which is created by using InterfaceBuilder.
Only a small set of the device functionality are considered. However, with these functions the usage of the prototype can be described in general.

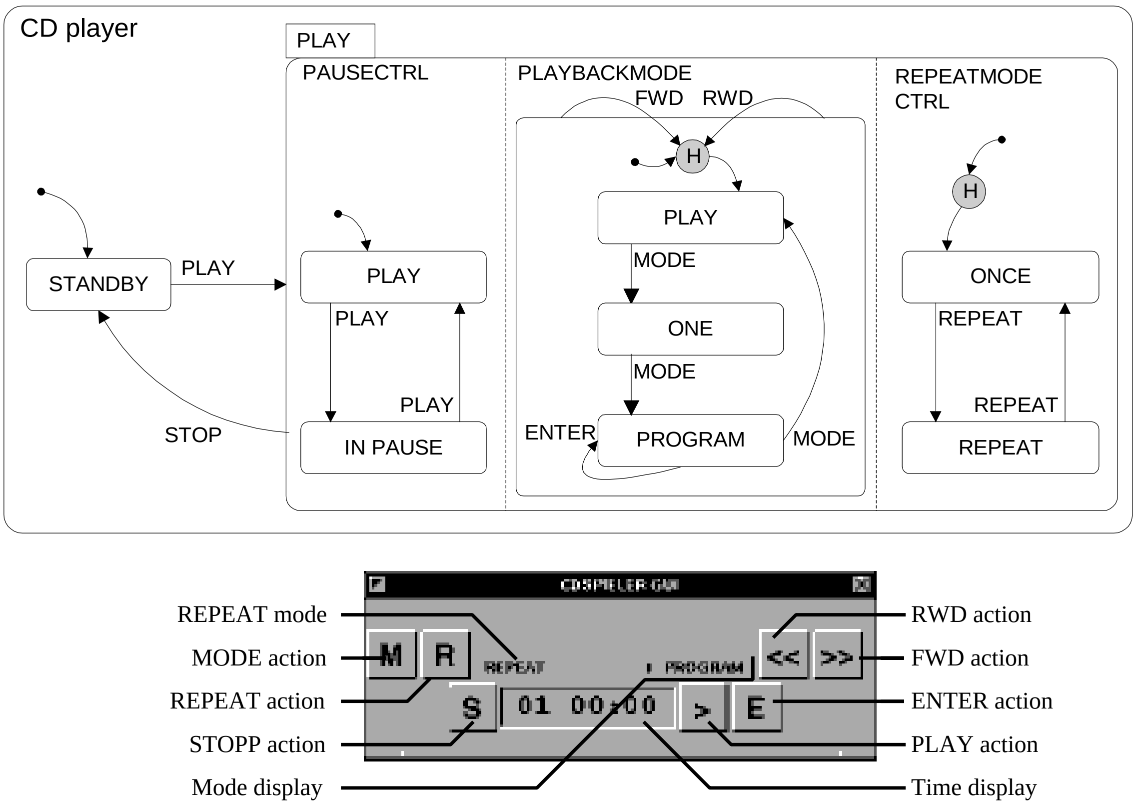


Fig. 4: Statechart modelled with Statemate and the related user interface of the CD player

The statechart models the individual states of the CD player and the possible transitions. Within the playback mode three different modes exist (PLAY, ONE , PROGRAM). Independently of these modes it can be selected between simple (ONCE) and repeated playback (REPEAT). A further parallel state controls the pause mode. The buttons on the user interface are marked with the initial letters of the transitions (STOP, ENTER, MODE and REPEAT) as specified in the statechart. The internal messages FWD and RWD are implemented not in the statechart, but in functionality.

6 Conclusions and future work

In this work the development environment StateWatch is presented, which consists of available tools and developed extensions. The environment offers support according to the task analysis and the specification of functionality within the development of the dialog control and a graphical user interface as well as a normative user model. The result is an executable prototype. The operation can be simulated by the normative user model.

As a special feature of this environment can be outlined that the development of the different models can be done without editing program code. The dialog control is specified by Statecharts in a graphical editor. The graphical user interface can be designed with InterfaceBuilder. The development and modification of a user model occurs with an editor, that is integrated in InterfaceBuilder. The goal of StateWatch is to show that an integration of normative user models can simplify the evaluation of technical systems.

In the future some extensions of the system are planned. These concern the further automation of the development process. One point is the automatic generation of the user model from the interaction specification of the Statecharts.

7 References

[1] ISO 13407 - Draft: *User centred design process for interactive systems.* International Organisation for Standardisation, Genf, 1998.

[2] ISO 9241-11: *Ergonomic requirements for office work with visual display terminals - Guidance on usability*. International Organisation for Standardisation, Genf, 1997.

[3] ISO 9241-10: *Ergonomic requirements for office work with visual display terminals - Part 10: Dialogue principles*. International Organisation for Standardisation, Genf, 1996.

[4] Marrenbach, J: *Modelling Complex Systems using Statecharts applied to the User Interface of an Advanced Flight Management System*. In: Proc. of the IEEE International Conference on Intelligent Engineering Systems INES´98, Vienna, Austria, pp. 43-48.

[5] Harel, D.: *Statecharts: A Visual Formalism for Complex Systems*. In: Science of Computer Programming, North Holland, Elsevier Science Publishers, Vol. 8, pp. 231–274.

[6] Kraiss, K.F.: *Modellierung von Mensch-Maschine Systemen*. In: Verläßlichkeit von Mensch-Maschine Systemen. ZMMS-Spektrum, Vol. 1, Berlin, 1995, pp. 15-35.

[7] Marrenbach, J.; Leuker, S.: *Konzept zur Evaluierung technischer Systeme in der Entwicklungsphase*. In: ITG-Fachbericht, Vol. 154, 1998, pp. 103-112.

[8] i-Logix Inc.: *Statemate Magnum - User Guide*. Andover, MA, 1997

[9] Apple Computer, Inc.: *Discovering OPENSTEP: A Developer Tutorial*. 1997.

[10] Card, S.; Moran, T.; Newell, A.: *The psychology of human computer interaction*. Lawrence Erlbaum, 1983.

[11] Kieras, D.; Polson, P.: *An approach to the formal analysis of user complexity*. In: International Journal of Man-Machine Studies, Vol. 22, 1985, pp. 365-394.

[12] Irving, S.; Polson, P.; Irving, J.: *A GOMS Analysis of the Advanced Automated Cockpit*. In: Proceedings of CHI ´94, 1994, pp. 344-350.

Towards the Generation of Tutorial Courses for Applications

García, F.

Escuela Técnica Superior de Ingeniería Informática
Universidad Autónoma de Madrid (UAM)
Cantoblanco, 28049
Madrid, Spain
Federico.Garcia@ii.uam.es

ABSTRACT In this paper we describe the underlying ideas of a system for the development of tutorial courses for interactive applications. This environment makes use of the user interface design technology based on declarative interface models. In particular, it uses hierarchical user task models to provide the designers with the possibility of generating tutorial courses about how to use highly interactive applications. The aim is both to reduce the generation and maintenance costs for the design of these tutorial courses and to build more dynamic and powerful tutorial course systems. The environment we propose is based on two separated modules. The first one, Advanced-HATS, is aimed to deal with the interactive teaching of isolated user tasks. In order to teach users how to perform a particular user task, this module uses the information in the user task model and the information provided by the underlying user task management system at run-time. The second module, the *scenario execution module*, is in charge of preparing the adequate *scenarios* for the first module to teach the users. Finally, CACTUS, a prototype tool implementing these ideas, is introduced.

1. Introduction

Current applications enclose a complexity that was unthinkable not many years ago. These systems, whose complexity is a consequence of the capabilities offered by the modern graphic user interfaces, have in many cases hundreds of commands, often operating in a different way depending on the context. This causes that only very advanced users can take advantage of a high percentage of the power of these systems. However, these tools rarely come together with systems that let the novice users systematically learn how to use them . This user profile is the focus of this paper.

We introduce the underlying ideas of a system aiming to palliate the problems related to current guidance components of software systems. It offers an architecture to design completely dynamic interactive tutorial courses. These courses can follow-up the activity of the pupils during their teaching sessions and act consequently depending on the user actions. Moreover, there is a drastic reduction in the costs of designing and maintaining the tutorial courses.

The system we propose lies upon the programming technology based on declarative user interface models (Szekely 1993), that provides many benefits in the development of applications (Puerta 1998), like partial design automation, validation of the solutions, and so on. The information in these models makes it possible for application-external tools to reason about the state of the application at run-time, in order to modify the interface behavior. In our particular case, the use of these models makes it possible to give the final users, that is the pupils, the possibility of receiving feedback from the continuous following-up of their activity and the evaluation of their knowledge on the tasks taught through the tutorial course.

We propose a system based on two modules. The first one, Advanced-HATS, is a tutoring system in charge of teaching user tasks. This subsystem provides users with context dependant dynamic messages in addition to several other innovative tutoring capabilities, such as context dependant graphical feedback. The second module, the *scenario execution module*, is in charge of preparing the appropriate contexts for the teaching sessions proposed in the tutorial course.

These ideas have been implemented in CACTUS, a prototype for developing courses. This system has been tested in the generation of tutorials courses for several applications. In particular, it has been proved with interactive interfaces for continuous digital simulation of systems such as the solar system and Volterra equations (Alfonseca 1998). It has also been tested in the generation of a tutorial course for the OOPI-TasKAD application, prototype for an object-oriented computer aided design environment based on the prototype/instance paradigm. Furthermore, tests have taken place on an electronic agenda that allows the typical options of this kind of tools. Finally, our tutorial course generation tool has been tested to generate some tutorial courses for *Schoodule*, an application that helps to generate school schedules.

This paper is structured as follows. First, we will summarize the problems associated with current tutors in the software context and the most relevant research projects in this field. Next, we will introduce the architecture of the system, including ATOMS, the user task management system we base our architecture on, the Advanced-HATS tutoring system and the *scenario execution module*, using examples to illustrate their operation. After that, the CACTUS tool will be briefly described, followed by the conclusions and future research lines of this work.

2. Context and Related Work

Even today, in most of the cases, the guidance component that complex applications offer to their users is based on hypermedia explanations. These components describe how each application command can be accessed and which functions they are supposed to perform. Obviously, this format has serious drawbacks related to their contents (Contreras 1998), that limit the usefulness of that kind of help system:

- The explanations are given at a very low level. The messages are given as a set of recipes, interconnected through hyperlinks, referring only graphic interface objects, and they never refer to higher conceptual level tasks. As a consequence, the explanations are not user task-oriented, but they are widget-oriented.

- Isolation of the help component. There is no interconnection between the application and the help system. Thus, the help component can not communicate with the user by accessing the application user interface.
- The help has a static style. The processes are fully explained before the user begins to perform the task, having her/him to switch between the help window and the application one to read the explanations and perform the task in parallel.
- There is no feedback from the help system towards the user, to indicate her/him when any of the steps s/he is accomplishing is not correct according to the system indications. Thus, the help system is completely passive and unable to correct or realize whether the user has understood its explanations.

Very few applications deliver, along with them, interactive systems to guide the users in the application learning. In those cases when a tutor for an application is included, there are very important drawbacks that make its usefulness/cost proportion too low to be produced for applications that will not be of wide use:

- Huge development costs of the teaching tool. The tutors usually simulate the application and the users have the impression that they are learning the real application, not just a simulated one, and the creation of that copy usually implies very high costs.
- Distance between the real work and the work with the tutor. Current teaching tools do not usually permit the learning using the work context of each user, but they only teach by means of predefined examples.
- High maintenance costs. Usually, each change in the application should be reflected in the tutor to guarantee the soundness between the application and what the teaching tool explains.

In the last years a great effort has been devoted to generate help for interactive applications using different paradigms. Next we will describe how these projects have surpassed some of the drawbacks mentioned above.

Cartoonist (Sukaviriya 1990) is a system based on the UIDE environment (Foley 1988) of generation of user interfaces based on declarative models. This system uses animations to communicate the information to the user. Cartoonist uses backchaining to navigate through the preconditions and postconditions of the commands specified in the model. The information this system produces is not hierarchically organized, so the explanations lack of the structure provided by hierarchical descriptions in terms of goals and subgoals.

On the other hand, the work of Pangoli and Paternó (Pangoli 1995) was pioneer in offering help from user task models. Thus, the information the user receives has greater semantic power than the one received from current help systems. However, it does not follow-up user activity, and then it does not offer her/him feedback about the task accomplishment.

Teach me While I Work, TWIW (Contreras 1996) also produces task-oriented help, but it goes a little beyond a help system and provides teaching capacities. In this sense, not only it informs users about the steps they should follow to accomplish a task, but it also filters user actions that are not correct according to the indications provided. One of the biggest problems of this system is that the task model that it uses does not include any contextual information, and it allows very limited types of sequencing among tasks. Moreover, this system is mostly oriented to users who have some familiarity with the applications they want to learn, but it is not intended to be used by novice users.

Finally, Help for ATOMS Task System, HATS (García 1998a), is a help system that uses user task models in order to generate the explanations. It has been later endowed with similar teaching capabilities to those of TWIW, as we will see when describing the Advanced-HATS tutoring system. In particular, HATS provides the user with dynamic help, updating the help messages according to what s/he needs to perform at each moment. The most novel feature of this system is its ability to offer graphical feedback, by highlighting application graphical objects, about what has been previously made and what is still to be done. HATS is based on the task models used by ATOMS (Rodríguez 1997, García 1998b), a system that manages user task models at run-time and allows the tasks to incorporate and to manage contextual information, in the form of parameters.

3. Architecture

The general architecture of the tutorial course generation system introduced in this paper is shown in Figure 1. As we can see, our proposal takes advantage of a refinement of the HATS help system, the Advanced-HATS tutoring system, and a scenario *execution module*, to provide the course designer with an environment with which, at the lowest cost, s/he can provide users with advanced interactive courses on their applications. The tutoring is oriented to teach the tasks that can be carried out in the applications, while the environment allows the tutorial course designers to specify the previous *scenarios* adapted for the teaching of those tasks. Both modules, Advanced-HATS and the *scenario execution module*, are built on top of ATOMS in order to get task-oriented support. In the next subsection, we will describe ATOMS in some detail.

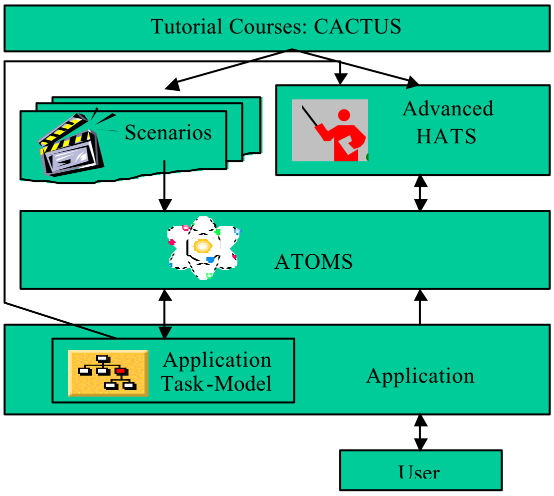


Figure 1: Architecture.

In the rest of the paper, we will illustrate the system features using examples from a tutorial course developed for *Schoodule,* an application that uses a database and a constraint solver to help on the generation of school schedules. The database can be described in a simple way by means of four entities and one relationship. The first entity corresponds to schoolteachers. The second one describes the different subjects that can be taken in the school. The information dealing with the groups of pupils is reflected in another entity, and the last one deals with the classrooms. Finally, the *assignment* relationship indicates which teacher imparts each subject for each group of pupils. From this information, the application automatically generates a schedule proposal, indicating both the timetable and the classrooms for each lecture.

3.1. ATOMS

ATOMS, Advanced Task-Oriented Management System, is a run-time user task management system. It is not a system aimed to design application user-interfaces from scratch, as TRIDENT (Vanderdonckt 1995) or ADEPT (Johnson 1995), but it is aimed to allow other processes to request to be notified when tasks are completed and invoke application tasks. These facilities support the constructions of agents that can assist users in various ways.

ATOMS based applications have to define their user task models, hierarchical representations, using a declarative language, specifying the tasks that can be accomplished with the applications and the rules establishing the relationships between different tasks. A detailed discussion of the advantages of hierarchical task models can be found in (Zeiliger 1997).

3.1.1. ATOMS Task Models

An application task model has to define two kinds of tasks: atomic ones and composed ones. The first ones model the interactions a user can directly perform with the application interface, that is, they describe the abstract interaction object associated to each atomic task. On the other hand, the composed ones allow modeling the top-level tasks a user has in mind when s/he carries out a certain set of actions. Both of them may have associated parameters that act as contextual information.

The task models, in addition to defining the application tasks, define the rules that hierarchically relate the tasks to each other. Each rule relates a particular composed task with a certain set of atomic and composed tasks. Furthermore, each rule includes other types of information. Some of them are:

- The execution relationship between the subtasks. This relationship can be:
  - 'sequence', where a subtask can not begin until every previous subtask has finished.
  - 'and', where the execution a of each subtask is independent of the execution of any other and everyone has to be accomplished.
  - 'xor', where executing any subtask implies that the task including them finishes.
- Description of the parameter flows between tasks. This description determines how the parameter values are obtained and how they are transmitted between tasks along the hierarchy.
- Which possible preconditions and postconditions must hold before and after a subtask execution.

3.1.2. ATOMS Internal Operation.

With respect to the internal operation of ATOMS, it is composed by four main blocks (see Figure 2): the *Dynamic Application Tasks*, the *Parsing Engine,* the *Task Modeling Tool* and the *Emulation* module. The first two modules follow-up the user activity while interacting with the application, in order to recognize at any moment which tasks s/he is carrying out. The third module is used for the interactive generation of the ATOMS task models. The last block executes tasks using animations, and it is provided as a service for external value-added tools to modify the application interface behavior at run-time. These four blocks will be analyzed in this section.

The first block, the *Dynamic Application Tasks*, represents both the tasks the user has accomplished throughout the work session (*Historic*) and the tasks s/he is currently carrying out (*Active Tasks*). ATOMS updates this information at run-time, as the user carries out the tasks specified in the task model. The main goal of the *Dynamic Application Tasks* module is to allow external value-added tools, such as Advanced-HATS and the *scenario execution module*, to reason about the application state in order to provide users with extra services. The second block, the *Parsing Engine*, analyzes the user interactions with the application. Taking into account the application task model and the state of the current tasks (*Active Tasks*), the *Parsing Engine* determines the state of the tasks, and updates the *Dynamic Application Tasks*. The role of the *Parsing Engine* is similar to the one played by *parsers* in *Natural Language Processing*, the lexical tokens being substituted by user interactions. Similarly, the Application Task Models play a role similar to *Unification Grammars* (Maxwell 1994).

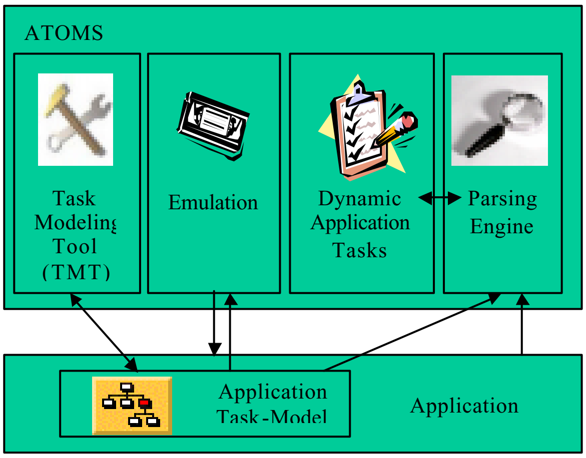


Figure 2: ATOMS architecture.

Task Modeling Tool.

The *Task Modeling Tool* (TMT) aims to palliate the greatest drawback of model based technology: the designer still has to model the application, so s/he has to get used to a new programming language. ATOMS offers TMT to allow the designer to specify the task-model once her/his interactive system has been built. This a-posteriori specification of the application user task model is a consequence of the run-time nature of ATOMS. This specification is done interactively, using both visual programming and programming by example techniques (Cypher 1993).

On one hand, TMT provides support to capture interactions and to generalize interactively them, thus allowing defining through examples the atomic tasks. The designer indicates the tool that s/he is going to perform an atomic task and s/he carries it out. Afterwards, s/he modifies it using a specialization and a generalization process. If the designer does not modify it, then ATOMS will consider that any interaction of the same type to the one provided will match. For example, if the designer has moved a graphical object, the interaction would be described as '*moving any object*'. An interaction modification includes two steps: the interactive selection of the set of relevant attributes from the interaction and the generalization of the values for those relevant attributes. The first step identifies which values ATOMS must have into account to match a user interaction against the one the designer is accomplishing. For example, the designer can indicate that the color of the object being moved is important, but the object size is not. The model designer tells TMT which attributes are relevant by selecting them from a list of attributes of the interactor object representing the interaction. This selection is a specialization of the interaction, since TMT forces the attributes to have the same values they had in the first example of the interaction. For example, if the moved object had red color, the interaction would be described as '*moving any red object*'.

The second procedure *generalizes* the relevant attributes by allowing the designer to repeat the interaction in a similar way. The system generalizes interactions using a rather simple algorithm that looks for a maximally specific expression representing a set of interactions (Mitchell 1982). For example, if in a second demonstration the user moves a green object, then the interaction will be described as '*moving any red or green object*'. After a demonstration, the system generates an expression including the values for the relevant attributes during each demonstration. The final representation of the interaction being defined, i.e. the description of its attached abstract interaction object, is referred to as an *abstract interaction pattern*. In the current TMT implementation, *equality* and *or* predicates are generated for each relevant attribute. A pattern represents the conjunction of all the expressions corresponding to each one of the relevant attributes for a particular interaction object. With just these two operators, TMT generates *abstract interaction patterns* in which each attribute value contains a disjunctive expression of equality tests. However, as ATOMS is able to manage general predicates, such as arithmetic and logic operators, more complicated expressions can be manually specified by advanced designers in order to declare *abstract interaction pattern* for atomic tasks.

On the other hand, TMT provides support to define interactively composed tasks and rules relating the tasks. As in the case of the predicates for the *abstract interaction patterns*, TMT does not allow to generate interactively every possible rule. The only expression for parameter transfer functions that TMT interactively generates is the identity function. Although this is a very common parameter flow function, advanced designers can manually specify several other functions, since ATOMS can manage them.

Emulation.

Finally, the fourth main module in ATOMS, *Emulation*, is entrusted with executing tasks using animations. This module is provided as a service to be used by external value-added tools in order to modify the application interface behavior at run-time. As we will see later, this module is the basis of the *scenario execution module*. During the execution of these tasks, a mouse-pointer appears moving along the screen in the same way a user would make it move by using the mouse to accomplish them. In addition to this, the same interim feedback and the same application behavior are obtained as if the user had accomplished the task. This module is capable of emulating the user interactions by analyzing the tasks and rules descriptions specified in the application task model. Starting from the model, the description of the task we want to emulate and the values of the parameters that are provided, *Emulation* decides which atomic tasks must be executed, as well as the corresponding parameter values and the order in which they must be carried out. Since each atomic task has an associated *abstract interaction pattern* that describes the corresponding interaction, *Emulation* analyzes the aggregation relationships of graphical and interaction objects in the target application in order to determine which steps to accomplish and with which values that interaction has to be performed. For example, this ATOMS module could be requested to execute the task *SelectTeacher* with the parameter *name* having the value *"John Hopkins"*, and it will execute it by means of animations, in the same way a user would accomplish it.

Emulation receives task execution requests in which the specified values for the parameter references parameter values for the top-level task, that is, parameter values of the composed task to be emulated. Since this module executes the top-level tasks by progressively decomposing them into subtasks to obtain the atomic tasks to be executed, one of the most important steps this module performs is transforming the values of the composed task parameters into simpler task parameter values. To perform this transformation, the inverses of the parameter transfer functions of the task model rules are applied, and the parameter values are propagated down along the task hierarchy. Thus, in the previous example, the textual parameter *"John Hopkins"* corresponding to the name parameter of the *SelectTeacher* task will be converted into a parameter, whose type is a graphical object and whose text content is *"John Hopkins"*. This label will make *Emulation* to search for a suitable graphical object during the execution of the *SelectTeacherFromList* atomic task, thus selecting the correct item from the corresponding list.

Emulation also admits task execution requests in which one or several of its associated parameters do not have any value assigned. For example, this module accepts requests to emulate tasks like *SelectTeacher*, with no specified parameter values. *Emulation* treats this kind of situation, in which the value of some parameter that has not been provided is needed, by indicating the user that s/he must provide the parameter value in an interactive way. In the cases in which it is possible, *Emulation* also offers feedback to the user on what possible values or options are available. In the previous case, *Emulation* has not been provided the teacher's name, so it does not know which element from the list of teachers must be selected. Hence, it makes every item in the list blink, and indicates the user that s/he must select one of them. From that moment on, the system only allows the user to select one item from the list, and disables the rest of the available interactions in the application. In general, not every parameter transfer function is bijective. Thus, when non-bijective functions are used to specify parameter transfer flows, it is not possible to convert a parameter value in its corresponding lower-level parameter values. This means that there are cases in which a certain composed task parameter value is not fully promoted to an atomic task parameter value. Of course, the selection of appropriate default values may be a suitable solution for providing inverse functions, but this would imply having more knowledge about the parameter semantics, something that would include undesired complexity into our task models. However, in many cases it is not a dramatic trouble, since *Emulation* will try to execute the task, independently of whether a particular parameter value has been successfully promoted or not. If any value needed to perform an interaction is not available, then the system will ask the user to provide it interactively.

3.2. From Tasks to Tutors.

In our environment, a tutorial course is composed by pedagogical units, each one teaching a set of tasks that are related by some criterion. Before teaching a task, the appropriate context for the teaching session may be prepared by means of the execution of a scenario.

As it has been already mentioned, we propose an architecture based on two modules (Figure 1). The first one, Advanced-HATS, is entrusted with teaching the user to accomplish a certain task. The second one, the *scenario execution module*, prepares the appropriate contexts for the teaching sessions, in order to make it possible to carry out the teaching of the tutorial course tasks.

The Advanced-HATS module is a refinement of the HATS help system (García 1998a), whose most important improvements are centered on the incorporation of tutoring capabilities to the helping capabilities already mentioned in section 2. In addition to HATS features, Advanced-HATS offers the possibility of acting with several degrees of flexibility. As a consequence, the system not only indicates the user how s/he must carry out a particular task, updating the explanations as s/he accomplishes the task, but it can also filter some wrong user actions, compelling her/him to follow a correct path. This filtering will be performed depending on which degree of flexibility is set. Another Advanced-HATS capability, not present in HATS, is the possibility of teaching a user how to perform parameterized tasks. This means that Advanced-HATS can give explanations related to the task parameter values in the messages conveyed to the user. For example, HATS is only able to teach tasks without any parameters, like '*Add a new teacher*', by indicating the user which interactions s/he has to perform and, in particular, in which text-input field s/he has to input the name. In contrast, the Advanced-HATS tutoring system is able to teach parameterized tasks like '*Add a new teacher with name John Hopkins*' by indicating that s/he has to input the '*John Hopkins*' text string in a particular text-input field. The tutoring system also filters actions that, although they are correct in order to accomplish the explained tasks, are not with respect to the values adopted by the task parameters, thus forcing the user to perform the task with the desired parameter values. In the previous example, it would filter any interaction aimed to input a name different from '*John Hopkins*'.

With respect to the *scenario execution module*, it prepares the necessary contexts to carry out the teaching of the tasks proposed in the tutorial courses. The *scenarios* are procedural descriptions, written in an specific programming language, of processes including references to application tasks, variables, conditional constructions, iterations, and so on. The *scenario execution module* interprets the corresponding procedural descriptions. For example, the unit in the *Schoodule* tutorial that teaches the user how to add a new teaching *assignment* to the application database previously prepares a scenario to add some default teachers, classrooms, and so on, to the database. In this way we can be sure that the user will not have any problem during the practice, for example if there is no teacher in the database to choose, or if there is no classroom in the database to select. The course designer may include in any unit a scenario followed by a task teaching session. Then, the execution of that unit will cause the system to ask the *scenario execution module* to prepare the *scenario* to change the state of the application. In our example this change adds some teachers, classrooms, etc. to the application database. After this, Advanced-HATS will teach the user how to add a new *assignment* to the database.

Task references in this language play the same role played by system calls in other environments: the interpreter, basing on the ATOMS' *Emulation* module, executes that task using animations. It is also possible to build *scenarios* interactively by performing the desired tasks in the application and telling the system to generate a scenario from those tasks. For example, the designer can perform the tasks for inserting some default teachers, classrooms, groups and subjects in the application database. After that, the designer can create automatically a scenario to perform those actions.

Nevertheless the environment allows users to practice the tasks in their own work context instead of using predefined *scenarios*, in order to provide designers with the biggest flexibility. Thus, the user can choose between three different behaviors: always execute the *scenarios*, never do it, or just choose whether to execute them or not. The fact of not executing a scenario may imply the impossibility of learning a particular task, because it may be impossible to perform a particular task in the user's current context.

4. The Implementation

CACTUS is a system for the interactive development of tutorials, based on the ideas explained before. It aims for releasing the tutorial designers from most of the intensive workload of developing the tutors.

CACTUS integrates designer oriented services, such as support for creating, editing, testing and debugging of tutorials, and pupil-oriented services, such as support for the execution of the courses. A course is composed by units, each one teaching a set of user tasks related by some criterion. CACTUS allows designers to create and modify interactively the tutorials, using programming-by-demonstration techniques, similar to the ones implemented in TMT.

CACTUS uses the metaphor of representing a tutorial as if it were an interactive textbook, and associates the most important parts of the tutorial with representative parts of a textbook. Most parts of the interactive books are automatically generated by CACTUS, and the designers only have to include the desired contents for the pedagogical units. The main advantage of our courses with respect to those generated with existing systems is the fact that our courses follow-up user activity. This makes possible to provide users with context-dependant explanations. Thus, the user does not learn by reading the book contents, but s/he learns by executing them and by following the indications they provide. This is completed by offering the use of predefined *scenarios* as context for the learning process, in addition to using the user's own work context. Another important feature is the automatic incorporation of hypertext links, so that it is easy to navigate through the books while the designer is released from explicitly providing the hyperlinks. In the rest of this section we will give a description of the structure of a CACTUS book.

First, CACTUS generates a cover like the one in Figure 3 (left). It includes the course title and a general description to provide an overall idea of the course purpose. Both fields are user-customizable.

Afterwards, CACTUS automatically generates an index of the course pedagogical units using the same order the designer follows to develop them, as shown in Figure 3 (right). Each entry contains a customizable unit title and a description of its contents. For each entry, a hyperlink is generated to the starting page of the corresponding pedagogical unit. The contents of a pedagogical unit will be explained later.

One of the main aspects of a CACTUS course is that it can be followed using several orders. In this sense, our interactive books are similar to those textbooks used in advanced courses, which usually have a predefined order to be read, but can be read using other orders, since the matters covered are complementary. CACTUS courses include the sequencing notion between the different pedagogical units of the tutorial. CACTUS automatically manages the sequencing, and permits a user to execute a unit depending on whether its previous units have already been taught or they have not. The ability to follow user interaction allows the system to manage automatically the unit sequencing. The next two parts of a book are narrowly related to this sequencing notion.

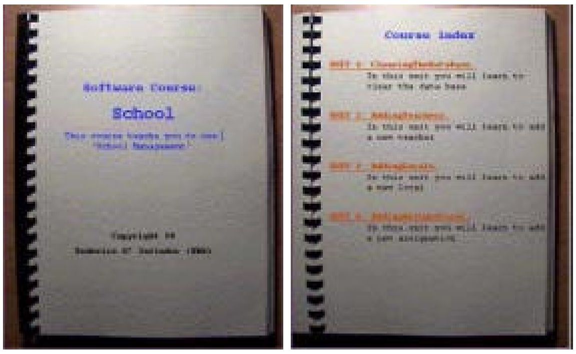


Figure 3: A tutorial book cover and index.

First, a directed graph representation of the course structure is generated, as shown in Figure 4 (left). Each node represents a pedagogical unit, and its color represents its state. The state of a unit indicates if it has been executed, if the unit has not been executed yet but it is accessible, or if the unit is currently blocked because some of its predecessor units have not been accomplished yet. An arrow from node A to node B indicates that unit A has to be performed before unit B. The states of the units are automatically updated by CACTUS as the user completes the course.

Second, a proposal of a sequencing of units for the course is generated, as shown in Figure 4 (right). In this proposal, when a node is visited all its predecessors have already been visited, so that it will never be the case that the user tries to access a blocked unit.

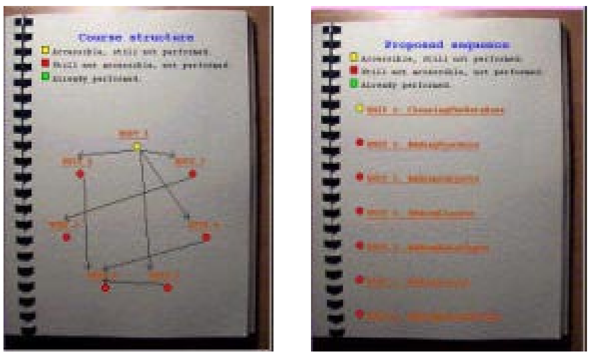


Figure 4: Tutorial book graph and proposal.

Afterwards, as many chapters as pedagogic units in the tutorial are presented (Figure 5, left). Each unit incorporates some tutoring sessions and, probably, some *scenarios* to execute. The page layout includes explanations about the tutoring sessions, the *scenarios* to be executed and images. CACTUS automatically generates explanations indicating the users the different things to be done, during the execution of that unit, from the references of the tasks to be taught. Of course, this automatically generated explanations are fully customizable by the designers.

Finally, CACTUS automatically generates a task glossary (Figure 5, right) at the end of the book, where every task being taught in the course can be found. Each entry relates a task with the list of units containing tutoring about it, and a hyperlink allows to easily access those units. Then, the process of searching about a certain task is reduced to a look-up process and to follow a hyperlink. This glossary is transparently managed by CACTUS, so the designer does not have to worry about updating it.

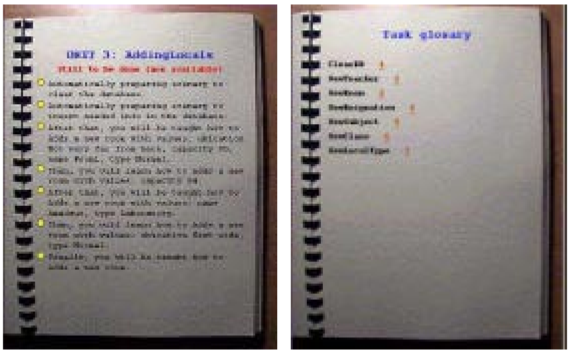


Figure 5: Tutorial book unit and task glossary.

4.1 Tutorial execution.

CACTUS allows executing some parts of the book to teach a user how to perform some user tasks using an interactive software application. The execution of tutorial courses can be also used by course designers to test and debug them at development-time.

Some sections of the book have a group of instructions associated. Instructions in this environment can be classified in three groups. The first one includes those for specifying task-teaching activities, preparing suitable *scenarios* to practice with, and showing users information about their performance. This kind of instructions has an immediate feedback on the application interface. The second ones do not have an immediate feedback towards the user, and includes low-level specifications such as variable assignments or instructions aiming to format the course (unit descriptions, unit titles, images, and so on). This group also includes control structures such as loops, conditional blocks or random execution blocks. Finally, a third group controls how the *scenario execution module* and Advanced-HATS operate, including setting the degree of flexibility for the tutoring module, the quantity of messages to be shown during the teaching activities or the *scenario* execution speed.

The executable sections of an interactive tutorial course book are the pedagogical units and the proposal section.

The pages of the pedagogical units usually contain task-teaching activities, examples of predefined tasks, *scenario* preparation instructions and feedback messages for the users. These kind of instructions are adequately related by the use of different types of control structures.

For example, in our course for *Schoodule* there is a unit that explains how to add a new teaching assignment. In this unit, the first thing the tutorial course performs is preparing a scenario. This *scenario* adds some default teachers, subjects, groups and classrooms. In this way, we are sure the user will not have any problem during the practice, because of a lack of previous information in the *Schoodule* database. Executing this first step of this unit will cause CACTUS to ask the *scenario execution module* to prepare the *scenario*, in order to add default information to the application database. Of course, the user may choose to use her/his own work context instead of preparing the *scenario* provided by the designer. After this, the unit will teach the user how to insert a new teaching assignment to the *Schoodule* database. This teaching activity will be done under the supervision of the Advanced-HATS tutoring component, which will explain the task to the user and will tell her/him whether her/his actions are correctly performed. Finally, once the user would have successfully inserted a new teaching assignment, the tutorial course will give the user some feedback about the task that has been just performed, by means of a feedback message that includes the parameter values of the new teaching assignment added.

When the user adequately accomplishes the pedagogical contents of a unit, CACTUS automatically updates the state of that unit, enabling the access to its succeeding chapters, if appropriate. When a unit is disabled, CACTUS will not permit pupils to execute it. CACTUS shows under each unit title its state: *'Blocked unit*, *'Already performed'*, '*Accessible unit'* or *'Being performed'*. During a unit execution, CACTUS remarks at each moment the instruction being accomplished, so that the user always knows how much s/he has already performed and how much remains to be accomplished.

As previously mentioned, the proposal page is also executable, and its execution means the successive execution, according to the order being shown, of the pedagogical units in the course. By executing the proposal page, CACTUS makes the unit precedence graph transparent to the user, so that s/he will never try to execute a unit whose execution is blocked.

5. Benefits and Future Work

We have described how the model-based user interface design technology can be used effectively to generate dynamic and interactive courses for software applications. The tutorial courses produced with this technology overcome many of the most important problems related to the current systems for tutoring generation.

We have summarized the capabilities of the Advanced-HATS tutoring system that make it a suitable tool for building tutorial courses. This module provides the user with dynamic tutoring, updating the tutoring messages according to what s/he needs to perform at each moment, and offering feedback about what has been previously done, through graphic references to the application interface. Thanks to this module, the system overcomes some of the main drawbacks in current teaching systems. The abstraction level of the explanations is higher, due to the hierarchical nature of the task models the system is based on. The tutor component is more integrated with the application, since it is able to reason about the application interface to provide graphical feedback. Moreover, the Advanced-HATS behavior provides a dynamic style to the explanations. Finally, as the system is based on a run-time task management system, it is able to follow-up user actions and to provide feedback about user activity.

We have also summarized the *scenario execution module*, a subsystem to provide the designers with support for preparing appropriate *scenarios* to practice with. This module also teaches the pupil by showing her/him how to carry out the tasks being performed, and its operation ensures the possibility of teaching the tasks of the tutorial.

We have described the architecture of ATOMS, the user task model run-time manager that provides our course generation architecture with task-oriented support. The use of user task models drastically reduces the maintenance costs of the tutorial courses, since it is almost reduced to maintain the models. We have shown how this task manager faces the generation of the task models by using interactive techniques. It would be very interesting to improve this specification by means of inductive techniques such as grammar induction (Maulsby 1997), or to integrate a more powerful machine learning algorithm with TMT, such as ID3 (Quinlan 1993), to generate more complex models.

We have introduced CACTUS, an implementation of an interactive interface to facilitate the access to the Advanced-HATS tutoring subsystem and the *scenario execution module*. This interface provides automatic support for building tutorials in a fully interactive way, reducing the development costs of the tutoring courses. This interface aims to integrate services oriented both to tutorial course designers and pupils, while isolating the direct access to Advanced-HATS and the *scenario execution module*.

Finally, we would like to port CACTUS to a client/server based platform, at least the part related with the course execution. Our aim is to facilitate the generation of multi-user distributed courses that can be simultaneously executed by several pupils under the supervision of a human tutor who could take decisions at any moment about the future execution of the courses. In this sense, some work has been already done. Thus, (García 1999) introduces a distributed implementation of the ATOMS task management system, which is oriented to manage workflows.

Acknowledgements

Special thanks to Roberto Moriyón for his extensive remarks on the ideas of this paper. This work has been partially supported by the Plan Nacional de Investigación, project numbers TIC96-0723-C02-01/02 and TEL97-0306.

References

Alfonseca, M., García, F., de Lara, J., and Moriyón, R. "Generación Automática de Entornos de Simulación con Interfaces Inteligentes", Revista de Enseñanza y Tecnología, ADIE, nº 10. December 1998.

Contreras, J. and Saiz, F. "A Framework for the Automatic Generation of Software Tutoring". In Proceedings CADUI'96, Computer-Aided Design of User Interfaces, Eurographics, Belgium, June 1996.

Contreras, J. “A Framework for the Automatic Generation of Software Tutoring”. Phd. Thesis, Université René-Descartes, Paris, 1998.

Cypher, A. “Watch What I do, Programming by Demonstration”. MIT press (ed. A. Cypher), Cambridge, Ma., USA, 1993.

Foley, J.D., Gibbs, C., Kim, W.C., and Kovacevic, S. “A Knowledge-based User Interface Management System”. Human Factors in Computing Systems, Proceedings CHI'88, 1988.

García, F., Contreras, J., Rodríguez, P. and Moriyón, R. “Help Generation for Task Based Applications with HATS”. In Proceedings EHCI'98, Creta (Greece), September 1998.

García, F., Rodríguez, P., Contreras, J., and Moriyón, R. “Gestión de Tareas de Usuario en ATOMS”. IV Jornadas de Tecnología de Objetos, JJOO'98, Bilbao, October 1998.

García, F. and Moriyón, R. “A Framework for Distributed Task Management”. Third Argentine Symposium on Object Orientation, ASOO’99. Buenos Aires, Argentina, September 1999.

Johnson, P., Johnson, H. and Wilson, S.: “Rapid Prototyping of User Interfaces driven by Task Models”. In Scenario-Based Design: Envisioning Work and Technology in System Development, J. Carroll (ed.), John Wiley & Sons, pp. 209-246. London, 1995.

Maulsby, D “Inductive Task Modelling for User Interface Customization”. Proceedings IUI'97: Intelligent User Interfaces. Orlando, Florida, 1997.

Maxwell, J.T. and Kaplan, R. M. "The interface between phrasal and functional constraints", Computational Linguistics, no.4, 1994.

Mitchell, T.M. “Generalization as Search”, Artificial Intelligence, 18(2), 203-226, 1982.

Pangoli, S. and Paternó, F. “Automatic Generation of Task-oriented Help”. In Proceedings UIST'95, Pittsburgh, ACM Press, 1995.

Quinlan, J.R. “Induction of Decision Trees”. Machine Learning, 1(1), 81-106. 1986.

Puerta, A.R. “Supporting User-Centered Design of Adaptive User Interfaces Via Interface Models”. First Annual Workshop On Real-Time Intelligent User Interfaces For Decision Support And Information Visualization, San Francisco, January 1998.

Rodríguez, P., García, F., Contreras, J. and Moriyón, R. “Parsing Techniques for User-Task Recognition”. 5th International Workshop on Advances in Functional Modeling of Complex Technical Systems, Paris (France), Julio 1997.

Sukaviriya, P. and Foley, J.D. “Coupling a UI Framework with Automatic Generation of Context-Sensitive Animated Help”, in Proceedings UIST’90, ACM Press, 1990.

Szekely, P., Luo, P. and Neches, R. “Beyond Interface Builders: Model-Based Interface Tools”. Proceedings of INTERCHI'93, 1993, pp. 383-390.

Vanderdonckt, J. “Knowledge-Based Systems for Automated User-Interface Generation: the TRIDENT Experience”. Technical Report RP-95-010, Facultés Universitaires Notre-Dame de la Paix, Institut d’Informatique, Namur. 1995.

Zeiliger, R. and Kosbie, D. “Automating Tasks for Groups of Users: A System-Wide “Epiphyte” Approach, in INTERACT’97 (ed. S. Howard, J. Hammond and G. Lyndgaard), Chapman & Hall Press, IFIP, Sydney, 1997.

Finding the Common Ground for Legacy Interfaces

Chris Roast, Richard Brophy and Andy Bowdin
Sheffield Hallam University
Sheffield, UK, S1 1WB
+44 (0)114 225 5555
c.r.roast@shu.ac.uk

Abstract:

It is well established within interface development that user requirements are conventionally identified and documented prior to design and implementation. With the growing uptake of information technology the development process also needs to manage the re-engineering of legacy systems. Re-engineering not only provides an opportunity to enhance user interface quality, but also to broaden the user base.

This paper reports on the techniques developed in the re-engineering of two distinct legacy systems with the aim of identifying and meeting common user requirements for both. More generally, it is proposed that these techniques should benefit user-centred design in the context of widening user bases.

1. Introduction

The growing uptake and awareness of information technology has provoked the proliferation of legacy systems. Developers are being required to re-engineer existing systems, which were considered perfectly adequate, but now fail to satisfy the requirements of more a sophisticated user base. In addition, to enhancing the system interaction, cross product re-engineering provides an opportunity identify and meet common user requirements more effectively.

User Interfaces for All encourages the recognition of the range and diversity of users for which interactive systems should be usable, focusing upon the non-exclusion of user groups. Within the context of interactive system development this not only concerns issues of effective and appropriate interaction but also the identification of the user tasks and requirements that are conventionally determined early in development (Ryan and Sutcliffe 1998, Carroll 1995). The work in this paper reports upon techniques that have been used to contribute to identify the user requirements common to distinct systems.

We report on the adaptation and application of Requirements Engineering (RE) and Human Computer Interaction (HCI) techniques to enable effective re-engineering for a broader user base. The work has been driven by a case study of a small software house engaged in re-engineering two legacy software products, with the aim of exploiting commonality between them. The specific legacy systems are the result of a history of poorly documented product evolution, hence, re-engineering activity can be classified as one of "design recovery" (Chikofsky and Cross, 1990) for the reverse engineering aspect of the project. The two products provide similar services to two vertical markets with distinct user bases. Hence, the activity naturally concerns itself with identifying and meeting user requirements that are common to the two user groups. The ideal outcome of the activity is to have an interactive system fulfilling the user requirements common to both groups and in addition satisfying the distinct requirements of a specific user group. We look towards maximising the core elements thus ensuring a degree of flexibility satisfactory for both existing user groups and new user groups.

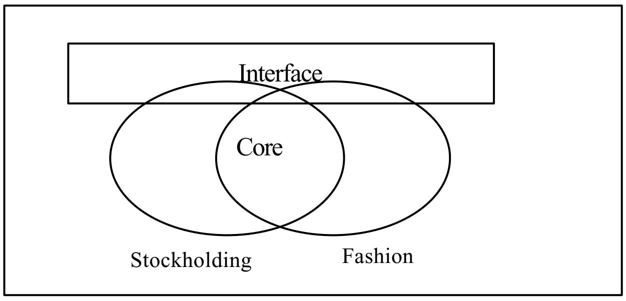


Figure 1. Stockholding and Fashion Systems areas of commonality

Below we further characterise the specific nature of the cross product re-engineering process and the selection of the techniques used (Section 2). Following this, the process adopted is detailed, emphasising the enhancements necessitated by the desire to maximise a broader user base (Section 3). The application of the techniques within the case study is described (Section 4). Finally we reflect upon the effectiveness of the techniques employed.

2. Selecting a Process

In addition to the object of maximising commonality between the existing systems, the adoption of a particular process was influenced by other development based concerns. In particular there was a desire to employ contemporary object-orientated and model-based techniques. Also, there was a need for flexibility within the process to enable adaptation to the specific constraints governing development, including: enabling re-engineering activity, closely matching existing development practices, and accommodating essential, though sporadic, access to users.

There has already been different approaches to enhancing legacy systems, Merlo et al (1995). Their process involved the creation of a meta graphical language from the character based interface. This meta language was then used to create a graphical user interface. Tools where used to make the process semi-automated. Our approach is different in two key aspects. First, that Merlo et al (1995) are looking at representation issues of an interface while we are looking at structural issues. Second, that they are not combining two legacy systems into one to extract the core functionality, but are translating one old system into one new.

Another method of requirements acquisition for an object orientated system would be 'Use Cases'. The 'Use Case' method can be described as a 'case of use' and is suggested by Booch, Rumbaugh & Jacobson (1999). There are problems with their application, if they are used on their own as a method of elicitation. Arlow (1998) pointed out that;

1. Use Cases can tend to trivialise business requirements, which means key requirements can be lost.
2. If functional requirements are being clumped together into one 'Use Case' then this can lead to a problem of prioritising later on in the project.

The acquisition of business requirements is one of the major reasons for reverse engineering as they are not available else where. When time and resources are limited then being able to prioritise key aspects of the development becomes crucial.

The process selected was the "Combined Model" developed by Kaindl (1998), this provided a process in which user requirements and interaction are clearly linked. More importantly, the model is accommodating of re-engineering activity and flexible enough to allow requirements to be developed and documented in a relatively opportunistic manner. Specifically, Kaindl's approach combines the use of goals, scenarios and functions, which enables requirements elicitation to proceed by developing coherent and complete inter-relationships between sets of all three of these concepts.

Despite the appropriateness of Kaindl's approach, it does not explicitly address our primary objective of identifying common requirements for legacy systems. However, the process offered by Kaindl is sufficiently rich and flexible to enable the consideration of identifying and maximising a common core.

The issues to be addressed in the adaptation of the Combined Model process:

- **Diverse Requirements** - How to combine diverse requirements associated with different user groups.
- **Legacy Features** - How to identify if characteristics of a function in the legacy system reflect valid user requirements, or if they are a design feature reflecting implementation details.

A more general requirement for the process adopted is that of its integration with existing development process, in particular its use within an object-orientated design process. The Combined Model integrates the documenting of scenarios and functions both of which can directly contribute to object-oriented development. However to help initialise and structure the process a task oriented framework based on Forbrig (1999) was employed. Forbrig encourages the development of inter-linked task model, object model and user model, for the existing system, the envisioned system and also for the application model.

3. The Adapted Process

The overall process is an adaptation of the Combined Model in which a common task model is developed to provide initial goals and scenarios. Given this information, inter-linked sets of goals, functions and scenarios are identifiable - the functions being a product of the analysis. In order to effectively address issues surrounding diverse requirements the process is enhanced to enable requirements to be classified as to their origin and appropriateness. To tackle the problem of legacy features the process is to focus upon deriving functions which are then compared with those evident within the legacy systems.

Preparation

Initially, user profiles of the two legacy systems were developed to clarify the context and nature of use. Following this, a common task model was developed, this represents the most specific account of task that can considered common to the use of both systems. For example, in order to find a stock details record one legacy system enabled search by "batches", where as the other system enabled search by "analysis grouping", hence, finding stock details was not decomposed any further. The common task model provides a reference to activities common to both systems and initial sets of goals.

Iteration

Kaindl's Combined Model describes dependencies between types of requirements, and articulates guidelines for an iterative process by which sets of goals, scenarios and functions can be developed and their consistency and completeness assessed. Two examples of the type of guideline:

If some goal is known, then try to link it to one or more known scenarios

If some scenario is known, then identify the goals that are achieved through it

(Paraphrased from Kaindl 1998)

Figure 2 shows the primary dependencies between these entities, in general:

- user goals can relate to system functional requirements by identifying the purposes of particular functions,
- scenarios can relate to functional requirements by identifying the behaviour relevant to particular functions,
- scenarios can relate user goals by considering the plans of activity.

Plans, purposes and behaviours have been classified as intersection entities to support the management of elements originating from different legacy systems. This provides an easy way of cross checking between functions, scenarios and goals to ensure that one legacy system scenario is not being solved by another's functionality.

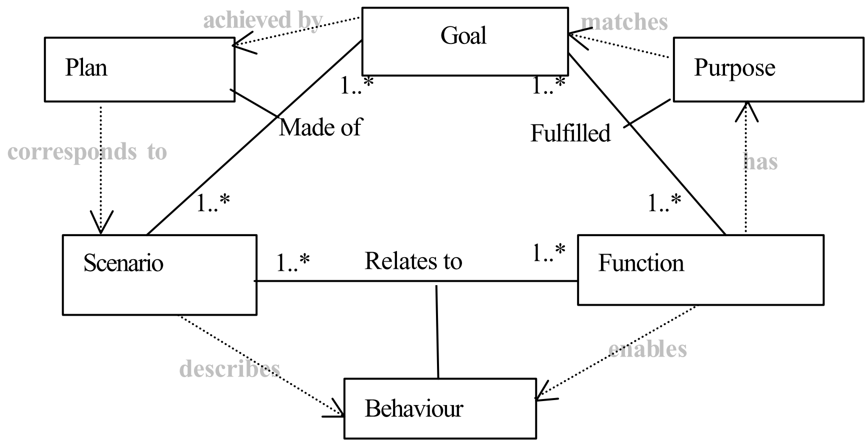


Figure 2. Adapted Model

In the above diagram (figure 2) the dotted lines and grey text relate to the original Kaindl (1998) combined model relationships between entities. They have been added to give extra clarity and to show the differences in the models.

Dealing with Diverse Requirements

Addressing the problem of maximising common ground between diverse requirements, each goal, scenario and function is classified as being of one specific legacy system or both. Given these classifications, inter-relationships such as the behaviour linking a function and a scenario, can be assessed in terms of whether common functions (or scenarios) are legitimate. In particular, we would expect to see a consistency in the treatment of goals, scenarios and functions of each class.

Identifying Legacy Features

The problem of focusing upon required functions evident in the legacy systems, as opposed to legacy design features, was tackled by comparing functions identified by the above analytic activity with those of the existing legacy systems. Any apparent omissions can be individually assessed as either legacy implementation dependant functions, in which case they were ignored, or required functions, in which case they were added to the function set. The importance of recovering a design by separating out implementation bias functions has been noted by Byne (1991). The overall process of consistency checking means that newly added functions naturally demand the identification (and possibly creation) of goals and scenarios to support them.

Validation

Throughout this process users can be involved in validating the requirements identified. However access to users for validation is an often limited, yet highly valuable resource. Hence, it is envisaged that the preparation phase should be the primary focus of validation activity, providing the information on which the subsequent analysis relies.

In ideal circumstance we would to ensure user involvement in the other phases of analysis especially in the assessment of legacy features and the generation and validation of scenarios, as demanded by the analysis. The Combined Model is of particular value here since it enables user validation based on differing requirements representations. User feedback can be gained on functions with a reference to situation specific examples, so a user can more easily relate the two.

Completion and Use

The iterative process of identifying and modifying requirements of each type continues until each set is considered to be internally consistent and complete and the relationships between each set are also coherent. For example, if there is function which is not described by any of the scenarios, the analysis may consider the legitimacy of the function or identification of a relevant scenario.

Once completed, the scenarios and functions identified could be employed as inputs to object orientated design. The scenarios can be used as a starting point for 'use case' analysis mapping into object orientated development. Functions can be used to identify object competence's (Booch, Rumbaugh & Jacobson 1999) as well as feedback to help prioritise the future development. While the Task model, used to create the goals, can be used as a basis for the design and construction of the interface.

5. Case Study

The case study is that of a small software house developing a new system combining two legacy systems. Although the systems are developed within the same company they are the product of separate and evolving product teams and have limited explicit or planned overlap. The systems have been developed, enhanced and maintained over several years.

The company culture would be defined as 'Club Culture' by Handy (1990). Club company culture can be characterised by being centred on the leader of the company and staff personal knowledge and experience. This means that the organisation is rich in personality. All software development has to take place in this context, as pointed out by Henderson-Sellers (1999).

The three main reasons the company is redeveloping its products are;

1. Efficient use of resources – At present there is duplication because of the two systems.
2. Market Trends – As noted in the introduction users are becoming more sophisticated.
3. Opening Markets – Identifying commonality will make it easier to create new niche market software.

The legacy systems are for the Fashion industry market and the ferrous and non-ferrous metals Stockholding market. Despite the two distinct markets, there is sufficiently common functionality for the processes of development and maintenance to benefit from identifying and modelling the common ground.

The Common Task Model

The preliminary phase involved was the creation of the user profiles and common task model, in both cases these were validated with members of the development teams familiar with the two user groups. The user profiles provided background material for

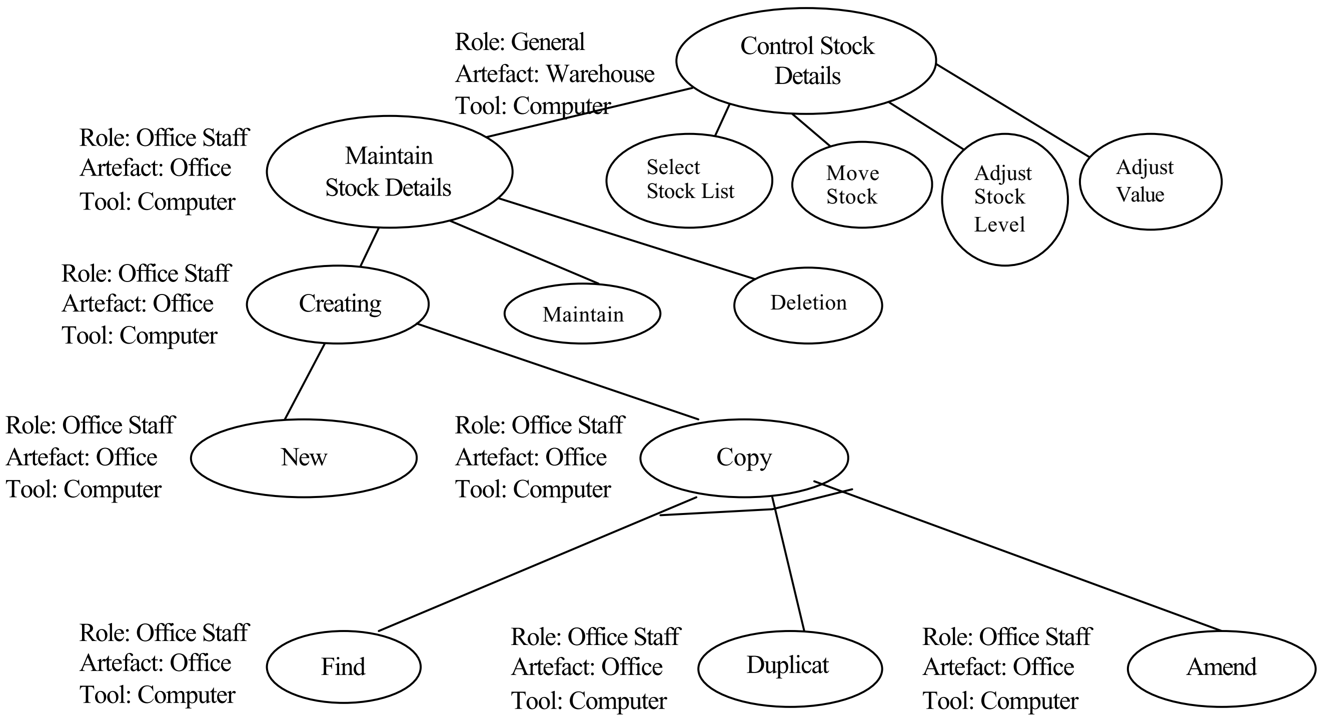


Figure 3. A Task Model

the development of the common task model. The task model employed a light-weight notation in which each task, the role, artefact(s) and tools were identified, conjoint or disjoint sub-task sets are positioned below tasks. Figure 3, illustrates one task within the common task model. Being common to both legacy systems, task decomposition was limited to task that failed to discriminate between the two systems. For example, within the figure 3, the task of "Find" is not broken down further since the selection criteria for finding in the Fashion system differed considerably from those in the Stockholding.

Using The Combined Model

The common task model provided the initial input for the use of Combined Model, in the form of an initial set of common goals. In addition, scenarios conforming with the task model, yet specific to a legacy system were elicited from users. These initial scenarios proved to be problematic, they were large and detailed, included aspects of legacy system behaviours and characteristics outside the focus of the re-engineering process. For example, while focusing upon say the stock control, a scenario may refer to packages, such as customer accounts, which was not subject to revision or analysis. These scenarios provided little scope for use in the Combined Model, and as a consequence it was decided that they should be segmented into smaller scenarios, and where possible, those which did not related to system being consider were excluded. These scenario segments served as a more effective initial set for the application of the Combine Model.

To assess and manage the origin and relevance of different requirements, each goal, scenario and function was tagged as follows:

B	-	Both	(Core System)
S	-	Stockholding	(Stockholding System)
F	-	Fashion	(Fashion System)

Hence, goals within the initial goal set were tagged B (since they were derived from the common task model), and scenarios within the initial scenario set were tagged S or F depending upon their origin. Figures 4 and 5 illustrate some of the scenarios and goals used.

The analysis within the Combined Model then proceeded by the iterative process of checking for consistency and coherence within the set of tagged scenarios, goals and functions, and refining the sets appropriately. This relied upon applying guidelines such as those of Kaindl illustrated in the previous section. However, in addition to developing the representations, we identified tag consistency rules to promote the identification of common requirements, while wishing to avoid the un-managed transference of requirements for one legacy system to the other.

Generalising over the analysis of the specific case study, the following tag consistency rules were proposed. The rational of these rules is of trying to find convergence in the two systems with a goal of closure. In general these have the effect of enabling common goals and functions within the analysis and encouraging system specific scenarios.

- **If a plan links, goal(s) X 'composed of' scenario(s) Y:**
  if goal(s) X is common (B), then its composed of scenario(s) Y can be both common (B) and specific to a system (S or F), e.g. G 1 B from Figure 6 is composed of S 1.4 B, S 1.3 B, S1.4 F & S1.5 S.
  if goal(s) X is specific to a system (S or F), then its composed of scenario(s) Y must also be specific to the same system (S or F).

- **If a behaviour links, scenario(s) X 'relates to' function(s) Y:**
  if scenario(s) X is specific to a system, then it relates to function(s) Y can be common or specific to the same system, e.g. S 1.4 F relates to F 1.7 F.
  if scenario(s) X is common, then it relates to function(s) Y which is common, e.g. S 1.7 B relates to F 1.3 B.

- **If a purpose links, function(s) X 'fulfils' goal(s) Y:**
  if function(s) X is specific to a system, then goal(s) Y can be common or specific to the same system, e.g. F 1.7 F fulfils goal G 1 B.
  if function(s) X is common, then goal Y is common, e.g. F 1.1 B fulfils goal G 1.1 B.

Only some of these rules were applicable within the case study, since not all circumstances arose. For example, the initial goals taken from the common task model were all classed as common (B), at no point did a goal specific to a system arise. Greater testing will be taking place in the future as the tag rules are applied in the rest of the systems elicitation acquisition. The first of the above rules is worth commenting on further. Since, both common goals and system specific scenarios formed the initial inputs to the process, to develop coherent and complete sets it was important to associate the two. This association is enabled by the first rule linking common goals to specific scenarios.

Complementing the above point the analysis also benefited from identifying more general scenarios based upon the common characteristics of more specific scenarios. For example, figure 5 shows scenario S.1.3 (B), as abstracting over S.1.4 (F) and S.1.5 (S).

At regular opportunities the representation of requirements in terms of goals, scenarios and functions was subjected to validation drawing upon the experience of developers familiar with the existing legacy systems and the two user groups. This activity served two purposes:

- The functions developed within the Combined Model could be compared with those known to be present in the legacy systems. In this way it was possible to address the issue of legacy features.

  One such case of implementation bias arose in the consideration of transferring data between modules of the system. The analysis of goals and scenarios motivated the requirement for a "transfer data" function. However the legacy system in fact provided two separate functions ("dump data to pool" and "get data from pool") closely mirroring an implementation detail.

- The overall integrity of the developing representation could be validated, in most cases recommendations and revisions can could simply be accommodated as additions or deletions of the elements of the requirement represented. As a consequence, the process could continue with little difficulty.

Goals

G 1	Create product details	B
G 1.1	To be able to create a product details record	B
G 1.2	To be able to make a duplicate of a product details record.	B

Scenarios

S.1.1	The end user is setting up the system, they already have all the stock codes structured and prepared beforehand. Data is entered in a batch.	B
S.1.2	The end user is creating a PO (Purchase Order)/SO (Sales Order)/ Quote and the item does not exist. It needs to be quickly created then and there to allow the creation of the document.	B
S.1.3	An end user will order from a supplier. They do not know the exact details until that item has been received and invoiced. This is because a container load is bought. The quantity is known but not the sizes or colours as this has to be checked on delivery. The quality of each item is not exactly known as their has to be a stock validation check when delivered.	B
S.1.4	The exact details needed, for S.1.3, are the size and colour	F
S.1.5	The exact details needed, for S.1.3, are weight and finish	S
S 1.6	There is a new item in product that is nearly the same as an existing item. It should be possible to copy the existing stock record and then amend the details of the new product details record in the specific area. If a code is entered that already exists then a copy of that record should be shown.	B
S 1.7	When creating a duplicate it should not be possible to create a new product without amending some detail on the new product because it will cause confusion if the only difference is the code.	B
S 1.8	The list of product details records should mainly show what records are still active, not deleted, stock details records. Though there are times when this information is needed e.g. when an item is being undeleted.	B
S 1.9	Miscellaneous notes about a product should be able to be entered as needed about the specific product.	B

Figure 4. Extracts from the Goal and Scenario sets

Functions

F.1.1 If all the stock details information is known when the record is being created then all that information can be put in from one screen. B

F.1.2 Notes about a product details should be able to be entered when needed on any screen that has anything to do with a product. B

F 1.3 The copy function will display the details of the old product details record into the new stock details record. It will not copy the code as that will have to be entered by the operator. An occurrence will only be added when the product details have been amended. B

F 1.4 If a duplicate code is created details for that code are shown. B

F 1.5 The create functions should be accessible at any time from other packages. B

F 1.6 If all the product details information is not known upon creation then the record can be created as a prospect record. This record can be used only as a limited product details record until all the needed information is entered or it is deleted. B

F 1.7 Size and colour should be able to be amended separately. The function should be accessible in other programmes. F

F 1.8 Weight and finish should be able to be amended separately. The function should be accessible in other programmes. S

F 1.9 The list generated of product details should be switchable so deleted items, prospect and active items can be turned on and off. B

Figure 5. An extract from the function set

Control Stock Details and Maintain Stock Details

Plan			Behaviour			Purpose	
Goal	Scenario		Scenario	Function		Function	Goal
G 1.1 B	S 1.1 B		S 1.1 B	F 1.1 B		F 1.1 B	G 1.1 B
	S 1.9 B		S 1.2 B	F 1.5 B		F 1.2 B	G 1.1 B
G 1 B	S 1.2 B		S 1.3 B	F 1.6 B		F 1.3 B	G 1.2 B
	S 1.3 B		S 1.4 F	F 1.7 F		F 1.4 B	G 1.2 B
	S 1.4 F		S 1.5 S	F 1.8 S		F 1.5 B	G 1.2 B
	S 1.5 S		S 1.6 B	F 2.2 B		F 1.6 B	G 1 B
G 1.2 B	S 1.6 B			F 2.3 B		F 1.7 F	G 1 B
	S 1.7 B			F 2.4 B		F 1.8 S	G 1 B
	S 1.8 B			F 1.3 B		F 1.9 B	G 1.2 B
	S 2.1 B			F 1.4 B			
	S 2.2 B		S 1.7 B	F 1.3 B			
	S 2.3 F		S 1.8 B	F 1.9 B			
	S 2.4 B		S 1.9 B	F 1.2 B			

Figure 6. Mapping goals to scenarios to functions and back to goals.

To position the tasks involved in the creation of the combined model within the requirements acquisition process and to give a holistic view of the process a meta model has been created (see Figure 7). The process starts with data collection, which included the creation of user

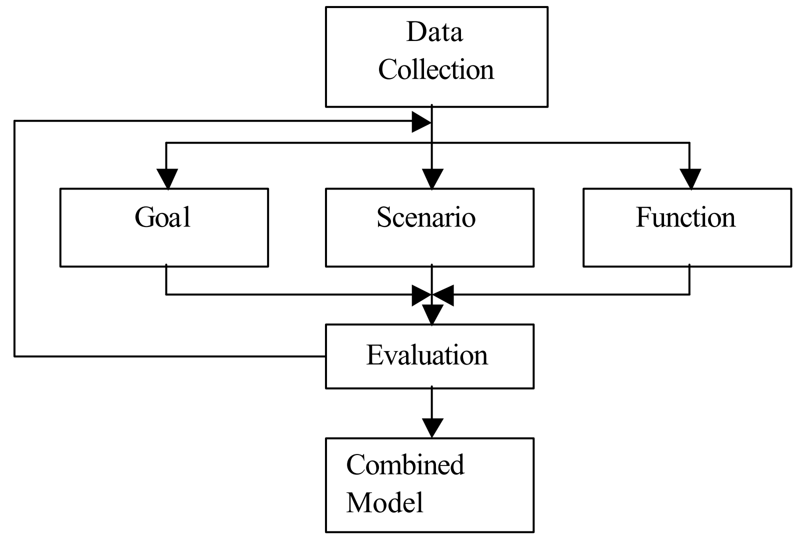


Figure 7. Requirements Acquisition

profiles and initial scenarios. In our case study we started with the creation of the goals, which included the creation of the common task model. Then segmented smaller scenarios were extracted from the initial scenarios. Last the functionality was extracted from the data collected from the legacy systems. This was evaluated and the cycle started again until all goals, scenarios and functions where combined and nothing was missing. The company culture does not support formal, rigid methods of elicitation. Hence the fact that the process is flexible and informal, supporting company culture and therefore gaining user support.

6. Discussion

The experience of applying the proposed process model for requirements analysis was governed by a wish to examine how the model performed as well as ensuring that appropriate decisions were made for the specific case study. Here we summarise these the departure from the envisaged process and consider their implication for other cross product re-engineering problems.

It was envisaged that the initial input for the requirements acquisition would be goals and scenarios drawn from the common task model. Despite the value of this initial goal set the process benefited greatly from legacy system scenarios which were specific to a single system. The scenarios could not be immediately articulated in terms common to both systems. In terms of the activities involved the scenarios provided for the analysis were motivated largely during the validation of the common task model. The analysis framework offered by the Combined Model could easily accommodate this form of input. On reflection the provision of specific scenarios complemented the goals evident in the common task model.

The nature of the scenarios employed during the analysis phase was considerably different from those envisaged. Although the role of scenarios is often to provide highly situated representations of how a system will be used, specific details appear to limit their applicability in the planned process.

But for these two points the planned process largely fulfilled our expectations. Following the spirit of Kaindl's Combined Model we prefer to identify practical descriptions of how the model can be used as opposed to prescribing specific approaches. Thus, we prefer to view these differences from our planned process as being indicators as to alternative ways of using the same model. For example, the guidelines for the Combined Model could be extended to reflect our treatment of scenarios, by including:

If two scenarios appear to have a general characterisation, consider including this as a more generic scenario.

Additional guidelines, such as this, represent further opportunities for the use of the Combined Model.

In terms of meeting the original objectives for the case study, the adapted process of analysis has served its purpose. The analysis identified and increased the set of common requirements crystallising functionality that was specific to the individual systems. The representation of common user requirements has driven the object based system development of re-engineered versions of the two systems, with a significant shared elements. The key features of the process which have contributed to this are the identification of functions prior to considering those of the legacy systems, and the use of representations tagged to reflect their applicability.

It has been noted that the description of the functions and the goals themselves are general and are not specific at all. This was done on purpose. The reason for this is that it was felt the very specific goals and functions would later constrict programmers. Which would have the effect of confining or constricting their ability to create software solutions to a function.

In addition to these features, the overall process benefited from minimising conflicts of ownership and priorities when considering common requirements from distinct sources. In the specific case study, the common task model appears to have offered a ‘starting point’ that was not biased to either legacy system. As a consequence, the task model appeared to facilitate co-operation between users in the subsequent validation activities. In particular the co-operative spirit was reflected in the developers of one legacy system requesting that specific functions become common.

Future

A series of similar re-engineering activities are to be carried out using the process described in this paper. This will enable the further refinement of the approach we have taken. Amongst the issues to be addressed in the future are;

- The treatment of non-functional requirements.
- Closer mapping of the requirements acquisition 'combined' model into the Object Orientated Design (OOD) phase of the project.
- Using the task model as a basis for the creation of the interface.
- Greater testing of the tag consistency rules.

7. Conclusion

The work described has been motivated by the desire to identify user requirements that are common to distinct legacy systems and use them as a basis for re-engineering those systems. The issues addressed in the activity are highly relevant when we come to consider User Interfaces for All and wish to consider identifying and meeting common user requirements. Clearly, a specific system is developed with a specific purpose in mind and as such will be distinct, however where there is commonality between systems, both prospective users and developers can benefit from the recognition of commonalties.

The process of requirements identification described in this paper provides one effective mechanism for identifying common user requirements, that is applicable in the highly relevant context of re-engineering of legacy systems.

Acknowledgements

This work has been supported by UK Teaching Company Scheme. In co-operation with PCI Systems and especially Andy Bowdin for all his help and support. The reviewers of an earlier version of this paper for their constructive criticisms.

8. References

Arlow (1998) - Jim Arlow, 'Use Cases, UML Visual Modelling and the Trivialisation of Business Requirements' Requirements Engineering, Vol. 3 No.2 Pg 150-152, Spinger-Verlag London Limited

Booch, Rumbaugh & Jacobson (1999). - Grady Booch, James Rumbaugh & Ivar Jacobson. The Unified Modeling Language, Users Guide, Addison-Wesley 1999

Byne (1991) – Eric J. Byrne, Software Reverse Engineering: A Case Study, Software – Practice and Experience, Vol 21(12), 1349-1364 (December 1991)

Carrol (1991) – John M Carrol, 'Designing Interaction, Psychology at the Human-Computer Interface', Cambridge University Press, 1991.

Carrol (1995) - John M Carrol, 'Scenario Based Design: Envisoning Work and Technology in System Development', John Wiley & Sons.

Chikofsky and Cross (1990) – Elliot J. Chikofsky and James H. Cross II, 'Reverse Engineering and Design Recovery: A Taxonomy', IEEE Software January, 1990

Forbrig (1999) - Peter Forbrig, Task and Object Oriented Development of Interactive System – How many models are necessary?, In DSV-IS 99 6th Eurographics Workshop Design, Specification and Verification of Interactive Systems.

Handy C.(1990) – C. Handy. Inside Organisations, 21 Ideas for managers. BBC Books. 1990

Henderson-Sellers (1999) –) – B. Henderson-Sellers, 'A Methodological Metamodel of Process', JOOP/ROAD pg 56-58, February 1999.

Kaindl (1998) - Hermann Kaindl, 'Combining Goals and Functional Requirements in a Scenario-based Design Process', People and Computers XIII, Hilary Johnson, Laurence Nigay and Chris Roast (eds), Springer-Verlag.

Merlo et al (1995) – Ettore Merlo, Pierre-Yves Gagne, Jean-Francois Girard, Kostas Knotogiannis, Prakash Panangaden and Renato De Mori, 'Reengineering User Interfaces', IEEE Software January 1995 pp 64-73

Ryan and Sutcliffe (1998) , M. Ryan and A. Sutcliffe 'Analysing Requirement to Inform Design', People and Computers XIII, Hilary Johnson, Laurence Nigay and Chris Roast (eds), Springer-Verlag.

Accessibility guidelines and scope of formative HCI design input: Contrasting two perspectives

C. Stephanidis and D. Akoumianakis
Institute of Computer Science (ICS)
Foundation for Research and Technology-Hellas (FORTH)
Science and Technology Park of Crete
Heraklion, Crete, GR-71110, GREECE
E-mail: {cs, demosthe}@ics.forth.gr

Abstract

The accessibility of interactive computer-based products and services has long been an issue of concern to both the Assistive Technology and HCI communities. Though progress has been slow, there have been several efforts aiming to document the consolidated wisdom in the form of general guidelines and examples of best practice. Despite their sound human factors content, these guidelines require substantial interpretation by designers before they can generate practically useful and context-specific recommendations. In this paper, we examine how different engineering perspectives in the implementation of guidelines may influence the quality of the final products.

1. Introduction

The term guideline is frequently used to refer to a general rule of thumb which applies for a particular class (or range of) computer artefacts (i.e., input/output device, window, text, etc). Usually, a guideline is a consolidated statement depicting existing design wisdom, or recommendations for which there is sufficient supporting evidence. Guidelines constitute a popular means for propagating human factors knowledge into the design and development of computer-based applications (Smith and Mosier, 1986). Their prime use is for formative design input. In the past, the human factors community has produced a wide range of guidelines for general-purpose software, specific interaction platforms (e.g., graphical user interfaces, the WWW) or specific application domains (e.g., aeroplane cockpits, educational software). However, despite their wide availability, HCI designers rarely use guidelines. This is due to several reasons, which have been extensively discussed, in the relevant literature.

To address some of these, researchers have attempted to provide tools for working with guidelines (for a review, see Vanderdonckt, in press). These are typically computer-based systems, which complement or substitute the prevailing paper-based guidelines propagation medium with more interactive means for accessing and applying guidelines. Their functional scope of such tools varies from making designers aware of existing guidelines and style guides reported in documents to (semi-) automatically interpreting and applying guidelines to a particular artefact. Representative examples of this class of tools are:

- EXPOSE system (Gorny, 1995),

- SIERRA (Vanderdonct, 1995),
- IDA (Reiterer, 1995),
- GuideBook (Ogawa, 1995),
- HyperSAM system (Iannella, 1994),
- Sherlock system (Grammenos, Akoumianakis and Stephanidis, 1999),
- DESIGN-AID system (Akoumianakis and Stephanidis, 1999), as well as
- the work by Henninger, Haynes and Reith, 1995, on experience-based usability guidelines.

1.1 Accessibility guidelines

In this paper, we will be concerned with a particular type of HCI guidelines, which are related to the design of accessible features to user interface software. Accessibility guidelines are typically documented on paper and reflect previous experience and best practice available for designing accessible interactive software. At present, there are several sources of guidelines for computer accessibility by disabled users (e.g., Thoren, 1993; HFES/ANSI 200, 1997; Rahman and Springle, 1997; Bergman and Johnson, 1995) as well as several World Wide Web sites[1] containing relevant documents and universal design principles (Story, 1998). These documents contain good design principles, design criteria, or design rules, which have been found useful, or applicable in specific application domains[2]. Examples of such applications can be found in Web browsers (e.g., AVANTI, Stephanidis et al., 1998), public access terminals[3], point-of-sale equipment[4] and cellular phones[5].

Despite the sound human factors input propagated through accessibility guidelines, a number of problems impede the use of such guidelines. First of all, accessibility guidelines, in the majority of cases, are expressed as general recommendations independent from context. Consequently, effective application of the guidelines requires substantial contextual interpretation – a task that is both interaction and collaboration intensive. Moreover, any interpretation effort is bound by the capability, experience and breadth of knowledge of the designer (or the specialist involved) regarding alternative access solutions, technical characteristics, etc.

Secondly, accessibility guidelines are seldom based on sound experimental grounds. In general, the available experimental work on assistive technologies is anecdotal (Casali, 1995), and does not cover the broad range of alternative solutions. Furthermore, due to the continuous radical changes in the mainstream Information Technology industry, some of the past experimental results rapidly become invalid or out of context. This is further complicated by the general lack of comprehensive evaluation methodologies and experimentation frameworks, which are necessary particularly in the context of people with disabilities.

Thirdly, accessibility guidelines are difficult to communicate to developers. Design input derived from guidelines is not always comprehensible or appropriated by the

---

[1] See for example http://www.w3.org/TR1999/WAI-WEBCON-TENT-19990505 and http://www.w3.org/TR1999/WAI-USERAGENT-19990331.

[2] It is also common that such guidelines are classified under disability groups (e.g., hearing impaired, motor impaired).

[3] See for example, http://trace.wisc.edu/world/kiosk/itms/prototypes/kiosk.html.

[4] See for example, http://trace.wisc.edu/world/kiosk/itms/prototypes/pos.html.

[5] See for example, *The Los Angeles Times*, Devices for the disabled are expanding access to modern communications, August 12, 1999.

development team. This is not only due to the typically demanding task of implementing these recommendations, but also due to the doubts that are frequently expressed regarding the validity of a particular recommendation in a given design case. A related issue is that of implementation expense. Typically, accessibility guidelines raise the cost of software development, as they demand substantial one-off programming efforts[6], which can seldom be reused across design cases. As a result, the cost of developing and maintaining the interface is high, and it increases dramatically with the number of different target user groups.

Fourthly, another commonly cited shortcoming of accessibility guidelines is that design recommendations derived through different sets of guidelines are frequently conflicting (sometimes, such conflicts may be encountered even within the same set of guidelines). In other words, one guideline may invalidate another guideline. At the same time, guideline documents or reference manuals offer no natural way of resolving ambiguities, which typically leads to arbitrary decisions by the design team.

Finally, the special vocabulary used by some of the available guideline manuals introduces an additional problem. This arises from the language used in these documents, which is not always comprehensible by the designers or the developers of user interfaces. As a consequence, additional training is usually required before the development team can effectively and efficiently use a guideline manual and implement the relevant recommendations.

1.2 The issue

Due to the above and perhaps, other reasons, accessibility guidelines are frequently neglected. As a consequence, the available products leave much to be desired in terms of end user perceived quality. In this paper, we seek to shed light to the gap existing between the human factors content of accessibility guidelines and the prevailing implementation/engineering strategies. It is argued that the inadequacy of currently adopted engineering perspectives, rather than the content or the quality of the guidelines, limits the impact of accessibility guidelines on HCI design. We attempt to do this by reviewing two practical case studies in which the same guideline is interpreted from different viewpoints. In the case studies reviewed, the common feature is the intention, on the part of the designer, to address a wider range of user requirements, including those of disabled and elderly people. Despite this commonality, however, there are differences underlying the various implementations, which impact certain quality attributes of the end products.

The paper is structured as follows. The next section provides an informative account of what accessibility guidelines are and how they can be translated into design heuristics and recommendations. Then we present alternative engineering perspectives for implementing accessibility guidelines and reflect on their relative merits. The paper concludes with a summary and discussion.

[6] This is also due to the lack of high level tools for building accessible user interface software.

2. Interpreting accessibility guidelines

To illustrate the point being addressed in this paper, we will make use of an example. The example covers accessibility of GUIs by motor impaired users. To this effect, a relevant accessibility guideline is as follows (Smith, 1996):

> G: "Provide a method for carrying out mouse, or other pointing device functions with the keyboard, or a keyboard emulator"

Interpreting such a guideline, gives rise to design heuristics, which are considered to be experience-based design rules. This means that design heuristics may not have been subject to empirical validation, as their applicability may be dependent on the context of use (i.e., user profile, task requirements, usage pattern, etc). A particular guideline may give rise to several design heuristics based on the interpretation of the guideline in a specific application domain, or context of use. Moreover, these heuristics may, or may not, be related to the same interface attribute and may, or may not, be of equal voting power. In cases where, for the same interface attribute, more than one design heuristics are applicable, then the designer typically needs to aggregate the competing alternatives to arrive at a plausible, maximally preferred solution.

For example, the guideline mentioned above may give rise to design heuristics at the level of the overall environment, or for specific user tasks. At the level of the overall environment the design heuristic may entail that:

> H1: "Switch access to Windows should be allowed for users with physical / motor impairments".

On the other hand, at the level of the user tasks, a design heuristic may entail that:

> H2: "Text editing facilities should be provided via a virtual keyboard accessible through a switch based interface".

Or, that

> H3: "Window management facilities should be made accessible through explicit function activation".

What is important to note is that heuristics may imply specific design artefacts which should be developed through a user-involved iterative process (Bannon, 1991) to reflect the requirements of the intended target user group(s). For instance, H2 implies a virtual keyboard such as that depicted in Figure 1, whereas H3 implies the toolbars of Figure 2, since a windowing platform may offer no programmatic control of window management facilities. In such a case, the examples depicted in Figure 2 illustrate tentative options for explicit (window management) function activation, which can be subsequently mapped to implemented versions on the desirable target platform, thus mapping heuristics to detailed design recommendations.

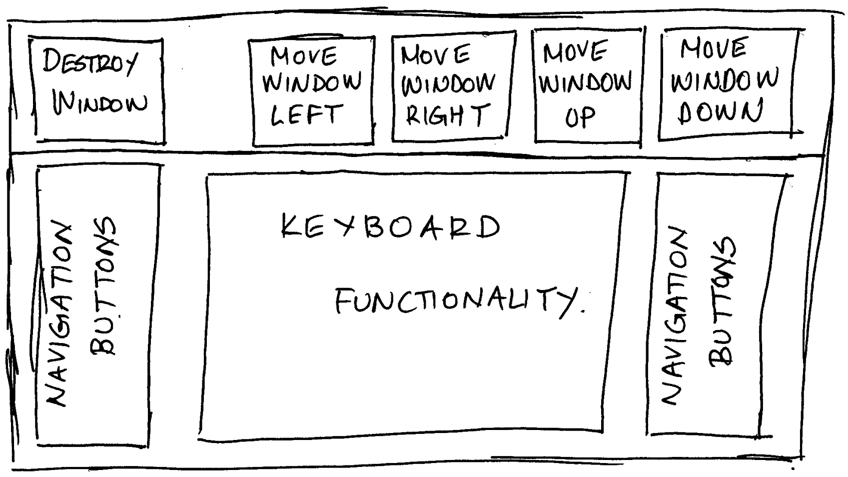


Figure 1: A paper mock-up of the virtual keyboard

Finally, recommendations reflect detailed design decisions which are usually related to implemented counterparts of design concepts, such as those depicted in Figure 1 and Figure 2. The scope of such decisions is typically bound to specific attributes of interaction (either at the physical level, or at the syntactic level). For example, recommendations relevant to the physical level of interaction may account for the "look and feel" of a virtual keyboard designed to facilitate text entry tasks for a user with moderate physical impairment, the choice of object classes, specific attributes of such object classes (e.g., the access policy for container objects, the topology of groups of items). Recommendations relevant to the syntactic level of interaction may determine aspects such as the dialogue command order that is to be used to accomplish specific tasks (e.g., Function-object versus Object-Function syntax), the function activation modes (e.g., explicit versus implicit activation), etc.

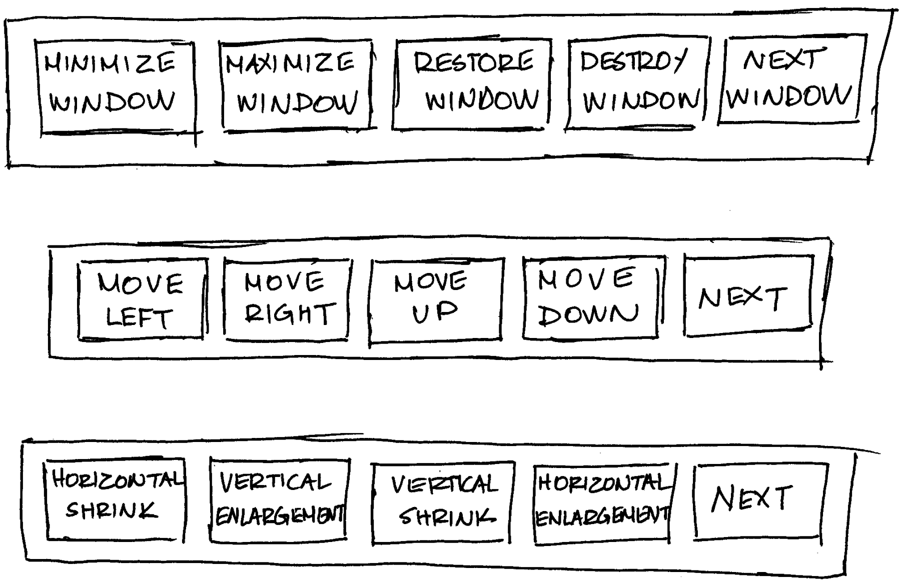


Figure 2: A paper mock-up of the toolbars for window management

It is important to mention that recommendations represent unambiguous statements about either the physical (lexical) or the syntactic level of an interface and, once derived, they can be directly implemented by a user interface developer. The emphasis upon the physical and syntactic levels of interaction is justified by the fact that accessibility typically affects both the choice of physical interaction resources (i.e., input/output devices, object classes and interaction techniques) as well as alternative dialogue organisation.

From the above, it follows that the main distinction between guidelines and recommendations can be summarised as follows. First, recommendations are derivatives of guidelines that occur through contextual interpretation of the latter into design heuristics, which in turn give rise to the recommendations. In other words, one guideline may give rise to one or more design heuristics which, in turn, result in collections of design recommendations. Secondly, design recommendations are not subject to interpretation. As such, they reflect concrete design artefacts, as opposed to abstract design patterns, and imply explicit assignments for those lexical, or syntactic items of a user interface that are needed to ensure accessibility by the intended target user groups.

3. Implementing accessibility guidelines: Two alternatives

Having distinguished between accessibility guidelines, design heuristics and recommendations, we will now examine two alternative implementation perspectives for the same guideline introduced earlier. The difference is in the interpretation of the guideline. It will be shown that analytical insights which broaden the scope of interpretation of a guideline may bring about substantial benefits for all parties concerned.

2.1 Implementing guidelines in specific applications

First, we examine how accessibility has been introduced in a word processing application running under a popular GUI environment. The example presents a typical case of mapping accessibility guidelines to specialised one-off design. Specialised design in fact aims either to provide enhanced interactive behaviours to visual constructions or to map them onto alternative modalities (with respect to the ones adopted in mainstream applications) based on some principles, design criteria or guidelines. It should be mentioned that the problem with such implementations is not the human factors content of the guidelines, which in the majority of cases is sound and can also constitute the basis for more generic solutions, but the engineering perspective that is adopted, which results in pre-packaged access features, hard-coded and difficult to maintain, upgrade, or map across different modalities and user groups.

Figure 3: An instance of the word processor GRAFIS for users with motor impairment in their upper limbs

As an example of specialised design, we briefly present the GRAFIS word processing system. GRAFIS runs under Windows and has been designed to be accessible by motor-impaired users[7]. The primary guideline which has been employed to provide access is that of replacing the mouse and keyboard operations with a switch-based emulation interface that enables the user to interact with the application through scanning. Based on scanning, several special dialogue elements have been introduced to facilitate specific tasks. Thus, for instance, text editing is facilitated through a virtual keyboard (see lower part of Figure 3), the elements of which are scanned by a highlighter in specific sequences. Selection is made by button press of a switch. Other navigation functions within the virtual keyboard dialogue are carried by corresponding switch-based operations. A typical dialogue with GRAFIS is illustrated in Figure 3. It should be noted that the scanning technique, as well as the special dialogue elements (e.g., virtual keyboard), have been hard-coded into the application, thus leading to a certain degree of inflexibility.

2.2 Implementing guidelines in the run-time environment

In the previous example, the prevailing approach to accessibility has been that of specialised design of an application so as to implement the necessary accessibility guidelines (e.g., emulation, switch-based access, off-screen model). It should be noted that guidelines may also be implemented based on analytical insights, thus offering more generic realisations of accessibility guidelines.

[7] The GRAFIS word processor has been developed at ICS-FORTH in the framework of the HORIZON ESTIA Project. This project ran from January 1996 to June 1998 and focused on the vocational training of unemployed disabled people, through the use of information technologies. Partners in the HORIZON ESTIA Project were: ICS-FORTH, University of Athens Department of Informatics, Idrima Kinonikis Ergasias, Panellinios Sindesmos Tiflon, Idrima Pammakaristos, Eteria Spastikon Voriou Ellados, Euroskills, INESC-Portugal.

In this section, we will provide an example of such a generic implementation. An alternative to implementing a separate, application-specific interface to an interactive system is to embed the accessibility features into the run-time environment, thus making it a property of the environment rather than of a specific application. In this manner, all applications running under that environment (e.g. Windows) may benefit from the corresponding accessibility features that are common and consistent throughout. For this to be possible, however, the run-time libraries of the environment should facilitate programmatic control of the available interaction elements and their attributes. Through such control, it would then be possible to augment the range and type of interaction techniques supported. This type of access possibility was originally demonstrated by the ACCESS project[8] and was subsequently taken up by mainstream vendors[9].

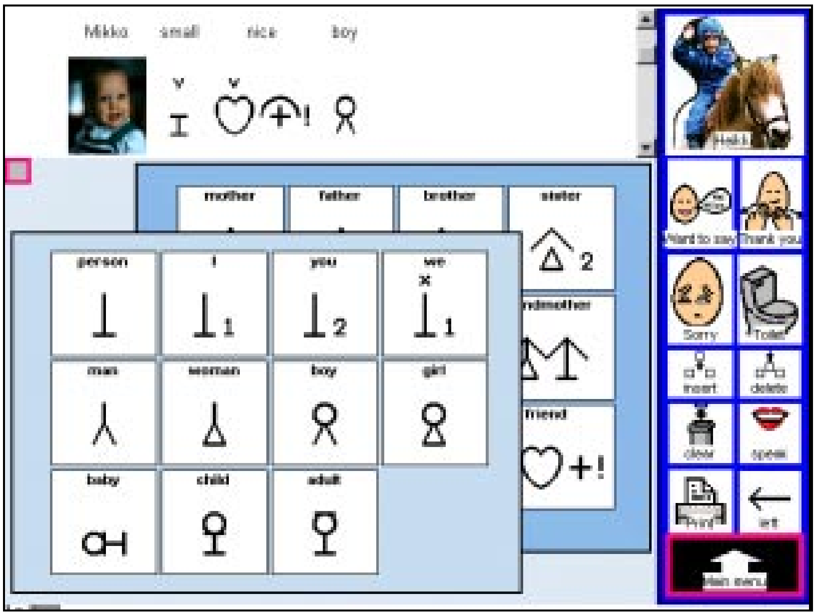


Figure 4: A screen view of a communication device, incorporating scanning, for a seven year old boy who communicates with Blissymbols.

The developments in the ACCESS project aimed to augment the Windows library of interaction objects with scanning. This should be contrasted to the example presented in the previous section, where scanning was fully programmed into the customised design. The augmented toolkit referred to as SCANLIB (Savidis et al., 1997b) provides developers with the capability to build full-functioning emulation-based interactive systems using the scanning technique. An example of an application (namely, a

[8] The ACCESS TP1001 (Development platform for unified ACCESS to enabling environments) project was partially funded by the TIDE Programme of the European Commission, and lasted 36 months (January the 1st, 1994 to December the 31, 1996). The partners of the ACCESS consortium were: CNR-IROE (Italy) - Prime contractor; ICS-FORTH (Greece); University of Hertforshire (United Kingdom); University of Athens (Greece); NAWH (Finland); VTT (Finland); Hereward College (United Kingdom); RNIB (United Kingdom); Seleco (Italy); MA Systems & Control (United Kingdom); PIKOMED (Finland). A description of the project is available on-line at the url address http://www.ics.forth.gr/~access/index.html.

[9] See for example Microsoft's Active Accessibility® (http://www.microsoft.com/enable/msaa/default.htm) and Java™ Accessibility API (http://java.sun.com/products/jfc/jaccess-1.2/doc/guide.html).

communication aid, Kouroupetroglou et al., 1996) making use of the augmented resources is depicted in Figure 4. What is important to mention about this implementation is that scanning is no longer hard-coded to serve the needs of a particular application (cf. GRAFIS); rather it is a property of the environment, thus facilitating the propagation of the same consistent set of accessibility features to all applications running under the same environment. It is, therefore, a typical example of a proactive account of accessibility, which is embedded and shipped as a generic environment-level property.

Sometimes, it may not be possible to provide generic access features by augmenting a particular toolkit. This is typical in cases where a non-visual modality is required, due to the user's lack of ability to articulate visual constructions. The problem that arises in such cases is that visual dialogue elements may not have a corresponding counterpart in the alternative modality; thus, one-to-one translations may not be possible or indeed meaningful. For instance, colour is an attribute of visual constructions, but it carries no meaning for artefacts with an auditory or tactile manifestation.

In such cases, an alternative to augmentation is to develop a toolkit comprising object classes, attributes and a collection of guidelines that allow developers to build applications with a generic target-modality manifestation. With such systems, the accessibility guidelines are embedded into the toolkit in the form of the toolkit's style guide. Thus, they offer another generic realisation of accessibility options relevant to the target modality. The first toolkit to incorporate such generic non-visual elements was COMMONKIT (Savidis and Stephanidis, 1995; Savidis and Stephanidis, 1998), which was also fully integrated as a target platform in the HOMER user interface management system. In subsequent years, the original version of COMMONKIT was refined and extended in the HAWK toolkit (Savidis et al., 1997a) which was used to provide generic non-visual access to hypermedia applications (Petrie et al., 1997) as well as Web-access (Stephanidis et. al., 1998).

3. Discussion and conclusions

From the example presented in the previous section, which are non-exhaustive but indicative of the type of prevalent articulation of accessibility guidelines, several discussion points arise. These are briefly elaborated below.

3.1 The human factors content of accessibility guidelines

The vast majority of the existing accessibility guidelines have been formulated on the basis of formative experimentation reflecting the results of best practice experiments. As a result, their human factors content is, in the majority of cases, sound. The problem in articulating these guidelines seems to arise from their context-independent orientation, which is inherited from the context-free research protocol of the human factors evaluation paradigm. It seems, therefore, appropriate to provide structured methods according to which a general guideline is decomposed and mapped onto context-specific heuristics, which in turn may provide the ground for valid recommendations. Such a structured method was briefly demonstrated in the examples presented in the previous section relating to the accessibility of user interface software[10]. The essential property of such a method is that it should be based on analytical insights informed either by a suitable theory (e.g. GOMS) or design technique (e.g. Design Space Analysis).

3.2 Reactive versus proactive practices and universal access

Since interpretation is the key issue when articulating accessibility guidelines, the next point of discussion is how to plan and structure such interpretations. Our experience demonstrates that the engineering perspective adopted determines the outcome of guideline interpretation. Thus, for example, the prevalent reactive perspective results in short term benefits which can be quickly outweighed by software updates and versioning. Such an approach does not suffice to provide the grounds for universal access in user interface software technologies. This is not only due to the radical rate of change in this industry but also to the short life cycles of existing products and the rate by which one technological trajectory overcomes a previous one. What is really needed is more generic accessibility solutions, which, however, can only be facilitated through more proactive engineering perspectives.

3.3 The cost factor

It is frequently argued that the main reason, which precludes more generic accounts to accessibility is the high cost which is involved. Our experience indicates that proactive implementations may have a design overhead, which, however, is fully justified by the resulting ergonomic benefits (e.g., consistency) and the substantial improvements in maintenance.

References

Akoumianakis, D., Stephanidis, C. (1999*): Propagating experience-based accessibility guidelines to user interface development*, Ergonomics - An International journal of research and practice in human factors and ergonomics, 42 (10), pp. 1283-1310.

Bannon, L. (1991): *From Human Factors to Human Actors: The role of Psychology and Human Computer Interaction Studies in System Design*, in Greenbaum, J., Kyng, M., (Eds.), Design at Work: Co-operative Design of Computer System, Hillsdale, NJ: Lawrence Erlbaum Associates, pp. 25-44.

Bergman, E., Johnson, E. (1995): *Towards Accessible Human-Computer Interaction*, in Nielsen, J. (Ed.), Advances in Human-Computer Interaction, Volume 5, 87-113, New Jersey, Ablex Publishing Corporation.

Casali, S. P. (1995): *A physical skills-based strategy for choosing an appropriate interface method*, in Edwards, A. D. N., (Ed.), Extra-ordinary Human Computer Interaction: Interfaces for people with disabilities. Cambridge University Press, pp. 315-342.

Gorny, P. (1995): *EXPOSE: An HCI-Counselling tool for User Interface Design*, Conference Proceedings of INTERACT'95, pp. 297-304.

[10] It should be noted that the same guidelines interpreted for a different context is likely to result in a different set of heuristics and physical recommendations.

Grammenos, D., Akoumianakis, D., Stephanidis, C. (1999): *Support for iterative user interface prototyping: The SHERLOCK Guideline Management System*, in S., Chatty and Dewan, P., Engineering for Human Computer Interaction, Kluwer Academic Publishers, pp. 299-318.

Henninger, S., Heynes, K., Reith, M. (1995): *A Framework for Developing Experience-Based Usability Guidelines*, Conference Proceedings of DIS'95, University of Michigan, August 23025, ACM, pp. 43-53.

HFES/ANSI (1997): *Draft HFES.ANSI 200 Standard. Section 5: Accessibility*, Santa Monica: Human Factors and Ergonomics Society.

Iannella, R. (1995): *HyperSAM: A management tool for large user interface guideline sets*, SIGCHI, 27(2), pp. 42-43.

Kouroupetroglou, G., Viglas, C., Anagnostopoulos, A., Stamatis, C., Pentaris, F., (1996): *A novel Software Architecture for Computer-based Interpersonal Communication aids*, in the proceedings of 5th ICCHP'96 (International Conference on Computers and Handicapped People), Linz, Austria, 17-19 July, pp. 715-720.

Ogawa, K., Useno, K. (1995): *GuideBook: Design Guidelines database for assisting the interface design task*, SIGCHI, Vol. 27, No. 2, pp. 38-39.

Petrie, H., Morley, S., McNally, P., O'Neill, A-M., Majoe, D. (1997): *Initial design and evaluation of an interface to hypermedia systems for blind users*, in the Proceedings of Hypertext97, Southampton, UK, 6-11 April, pp. 48-56, New York: ACM Press.

Rahman, M., Sprigle, S., (1997): *Physical Accessibility Guidelines of consumer product controls*, Assistive Technology, vol. 9, pp. 3-14.

Reiterer, H. (1995): *IDA: A design environment for ergonomic user interfaces*, Conference Proceedings of INTERACT'95, pp. 305-310.

Savidis, A., Stephanidis, C. (1995): *Building non-visual interaction through the development of the Rooms metaphor*, in the Companion Proceedings of 1995 ACM Conference on Human Factors in Computing Systems (CHI '95), Denver, USA, 7-11 May 1995, New York: ACM Press, pp. 244-245.

Savidis, A., Stephanidis, C. (1998): *The HOMER UIMS for dual user interface development: Fusing visual and non-visual interactions*, Interacting with Computers, 11(2), 173-210.

Savidis, A., Stergiou, A., Stephanidis, C., (1997a): *Generic containers for metaphor fusion in non-visual interaction: The HAWK Interface toolkit*, in the *Proceedings of Interfaces '97*, Rault, J-C., (Ed.), "La Lettre de l'IA" (The Advanced Information Technology Newsletter), Montpellier, France, 28-30 May, pp. 194-196.

Savidis, A., Vernardos, G., Stephanidis, C., (1997b): *Embedding Scanning Techniques Accessible to Motor-Impaired Users in the Windows Object Library*, in the Proceedings of the 7th International Conference on Human-Computer Interaction (HCI International '97), Salvendy, G., Smith, M., Koubek, R., (Eds.), San

Francisco, USA, 24-29 August, pp. 429-432.

Smith, J. W. (1996): *ISO and ANSI Ergonomic Standards for Computer Products. A guide to implementation and compliance*, New Jersey: Prentice Hall PTR.

Smith, S. L., & Mosier, J., N. (1986): *Guidelines for designing user interface software*, (Report No. MTR-10090, ESD-TR-86-278). Bedford, MA:MITRE Corp.

Stephanidis C., Paramythis A., Sfyrakis M., Stergiou A., Maou N., Leventis A., Paparoulis G., Karagianidis C. (1998b): *Adaptable and Adaptive User Interfaces for Disabled Users in the AVANTI Project*, in Trigila, S., Mullery, A., Campolargo, M., Vanderstraeten, H., Mampaey, M. (eds.): *Proceedings of the 5th International Conference on Intelligence in Services and Networks* (IS&N '98), "Technology for Ubiquitous Telecommunication Services", Antwerp, Belgium, 25-28 May 1998. Lecture Notes in Computer Science, Vol. 1430. Springer-Verlag Haidelberg Germany, 153-166.

Story, M. F., (1998): *Maximising Usability: The Principles of Universal Design*, Assistive Technology, vol. 10, pp. 4-12.

Thoren, C., (1993): *Nordic guidelines for computer accessibility*. The Nordic Committee on Disability, Stockholm: The Swedish Handicap Institute.

Vanderdonckt, J. (in press): *Development milestones towards a tool for working with guidelines*, Interactive with Computers. Special issue on Tools for Working with guidelines.

Vanderdonkct, J. (1995): *Accessing guidelines information with SIERRA*, Conference Proceedings of INTERACT'95, pp. 311-3-16.

Integrating Universal Design into a Global Approach for Managing Very Large Web Sites

Michael Cooper[1], Quentin Limbourg[2], Céline Mariage[2] and Jean Vanderdonckt[2]

[1]Center for Applied Special Technology (CAST)
39 Cross Street, Suite 201, Peabody, MA 01960 (USA)
Phone: +1-978-531-8555, Fax: +1-978 531-0192
E-mail: mcooper@cast.org, URL: http://www.cast.org

[2]Université catholique de Louvain (UCL), Institut d'Administration et de Gestion (IAG),
Place des Doyens, 1 – B-1348 Louvain-la-Neuve (Belgium)
Phone: +32-10-47.85.25 - Fax : +32-10-47.83.24
E-mail: {limbourg,mariage,vanderdonckt}@qant.ucl.ac.be
URL: http://www.qant.ucl.ac.be/membres/jv/

Abstract. Very Large Web Sites are a particular category of web sites where the potential of traditional evaluation process for ensuring usability is significantly contracted by the size of the web site. Since this kind of web site is authored, designed, evaluated, and maintained by a wide variety of people who have specific information demands over a huge set of web pages, we believe that Universal Design principles should be integrated into the current approach for managing such web sites. We propose to support Universal Design principles by integrating related guidelines into a global approach for managing Very Large Web Sites. This approach is supported by Extended Bobby, an extension of the Bobby tool that provides (i) evaluation on demand ; (ii) a repair tool that proposes to authors of web pages new HTML code fixing usability problems that Extended Bobby itself has merely identified and explained ; (iii) a usability site tracker that keeps track of usability problems of the web sites, automatically sends e-mails to authors with the repair proposal, and helps site managers and webmasters to manage the pages evolving in time.

1. INTRODUCTION

We hereafter refer to Very Large Web Sites (VLWSs) as any large-scale, information abundant, interactively rich web site installed in a distributed environment (e.g., in different physical locations) with heterogeneous software and hardware (e.g., on different servers). A wide range of persons typically maintains a VLWS. It is connected to a large set of databases and contains several thousands of web pages. For example, the Decathlon web site (www.decathlon.fr) is a 10,000 pages VLWS presenting users with a wide variety of information on sports articles and leisure. Similarly, the web site of Université catholique de Louvain (www.ucl.ac.be) contains 40,000 pages on research, courses, and activities on most domains of human sciences and is maintained by a potential range of 1500 persons supervised by 12 local webmasters and a general webmaster. The management of a large university web site is a demanding task as reported in [Nevile96]. Different types of actor typically participate in the design, the implementation, the evaluation and the update of a VLWS:

- *Document author* is any person who is responsible for writing and editing a series of web pages with appropriate tools such as word processor, document manager, HTML editor, converter (for example, a secretary);
- *Document responsible person* is the person guaranteeing the information contents of a document designed by a document author. This person could be a hierarchical supervisor or the document author him/herself (for example, a professor);
- *Site manager* is any person coordinating the web page publishing for any leaf node entity in the organization hierarchy (for example, the site manager of a department);
- *Local webmaster* is a person striving for the utility and the usability of web pages for related entities (for example, a webmaster for all departmental sites in a faculty);
- *Global webmaster* is the person coordinating the utility and the usability of the web pages for the whole VLWS (for example, a webmaster for all faculty sites in a university).

On one hand, document authors and responsible may talk different languages, may have various cognitive profiles and backgrounds and may have separate information demands. Although they do not necessarily have knowledge or experience in usability of web pages, they tend to prefer specific presentation styles and separate dialogue types for their own web pages to be quite different from what the others are designing. On the other hand, site managers, local and global webmasters are responsible for ensuring some form of usability and consistency across these web pages, thus introducing a counter-force. To fulfill their role, site managers, local and global webmasters currently follow a manual approach consisting in the following activities:

1. they regularly evaluate the set of web pages across a defined set of web design guidelines according to a heuristic inspection method;
2. they manually write a usability report where detected usability problems and guidelines discrepancies are documented;
3. they send the usability report to the document author or the document responsible person and ask them to solve the documented problems and to fix the discovered discrepancies;
4. they regularly remind document authors and responsible to take these considerations into account and they iterate the whole process.

These activities lead to the following shortcomings:

- due to the size of a VLWS, it is impossible to manually manage the above activities; therefore, the guidelines should be evaluated as automatically as possible and the usability errors should be reported by appropriate software;
- due to the lack of time or lack of interest, most document authors and document responsible persons devote little or no time to address the documented usability errors; therefore, some proposal should be produced by an interactive repair tool based on the usability errors that have been previously reported;
- due to the progressive appearance of new types of guidelines to be embodied in the evaluation, such as new design rules, guidelines from any custom corporate style guide, and new standards, involved people are rapidly blocked by the currently existing guidelines; therefore, the software should be open and flexible enough to extend the knowledge base of guidelines to be evaluated;

- due to the various types of actors involved, the different information demands, and the population diversity, guidelines from the domain of Universal Design should also be supported; therefore, the software should be able to accommodate this type of guidelines as document authors are rarely aware of them.
-

In general, Universal Design means designing services and resources for people with a broad range of abilities and disabilities [Stephanidis98,99]. Universal Design promotes equitable use [Coombs99], builds flexibility into the resource so that it can accommodate a wide range of individual preferences and abilities, is simple and intuitive, allows for duplication of information in several formats (e.g. written, spoken), and requires minimal physical effort [Connell97]. In particular, Universal Design for web sites means that a web site should be usable enough to accommodate a wide range of visitors having various information demands, having different cultural backgrounds, and equally important, having disabilities or not, limited computer facilities or not. This last issue is often referred to as the accessibility of web sites [Access98, Bergman95].

The goal of this paper is consequently to present a new global approach for managing a VLWS by integrating Universal Design and supported by appropriate tools. The rest of this paper is structured as follows: section 2 describes into more details the current global approach followed to manually manage a VLWS, its data flow, and reports on the shortcomings of this approach ; section 3 identifies the need for integrating Universal Design principles in this approach ; section 4 exemplifies how the Bobby software enables site managers or webmaster to evaluate a series of web pages across accessibility guidelines ; section 5 describes an extension of this software to support evaluation on demand, automated or computer-aided evaluation of guidelines ; finally, section 6 describes the proposed global approach for managing VLWSs being supported by the extended Bobby tool along with the repair tool and the usability site tracker, and its data flow. Section 7 concludes by presenting the expected benefits of this global approach with some future works.

2. THE CURRENT MANUAL APPROACH

During the maintenance of a VLWS in any organization, one or several methods can be used to evaluate its usability. In order to be concrete, we assume that a heuristic inspection method [Bastien95] based on guidelines will be used throughout the rest of this paper. In this variant, the general heuristics are replaced by a predefined set of criteria of guidelines to be assessed for each considered page of the VLWS [Bastien95,98]. Numerous sources provide such web design guidelines in general [Grose98, IBM97, Ratner96, Usable98, UseIt98, Yale97] and for accessibility and Universal Design in particular [Access98, Lowney96, Washington99].

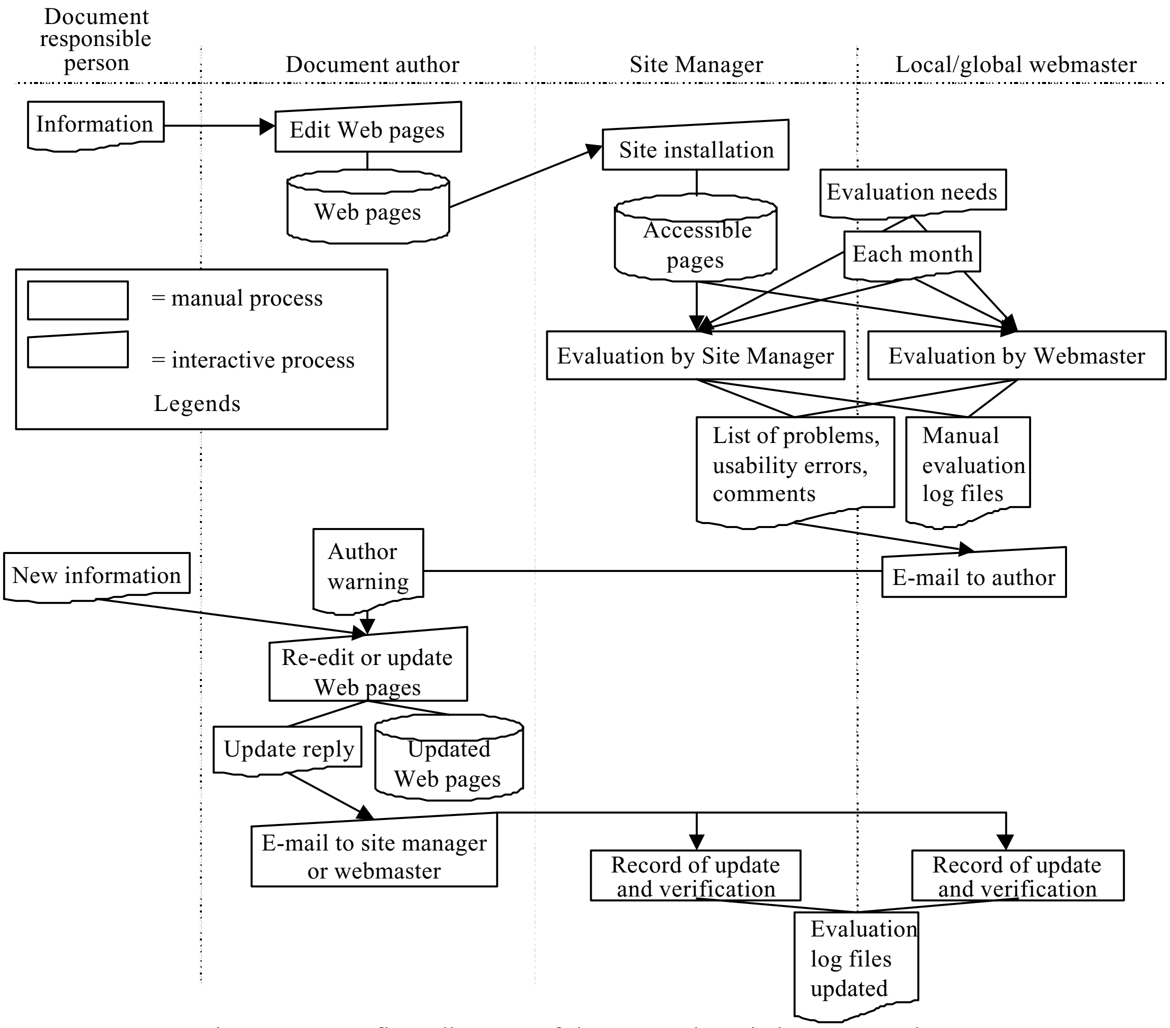


Figure 1. Data flow diagram of the currently existing approach.

The currently existing approach for conducting this VLWS evaluation is illustrated in fig. 1. Although we focused on a particular evaluation method, we believe that this figure will remain similar for any other evaluation method, whether empirical or analytical.

When someone (the document responsible person) would like to publish information on the VLWS, this person provides a document author with the information to be published, e.g., a document, a financial report, an information bulletin, or a list of references. The document author then edits web pages to put this information on line. HTML editors, document converters (e.g., from a word document format to HTML), development environments help the document author for this purpose. Since the pages will be integrated in the VLWS, they cannot be produced randomly : web design guidelines can express the rules that govern the presentation and the navigation of these pages. These guidelines basically come from five types of sources [Scapin90, Vanderdonckt99]:

1. Compilation of guidelines;
2. Style guides, whether they are general or specific;
3. Standards;

4. Design rules, as found for example in screen templates;
5. Ergonomic algorithms that automatically produce usable web pages.

For instance, a design rule can specify that each web page should be terminated with the name of the document responsible person and the clickable name of the document author (fig. 2).

[UCL] [Inventaire des recherches] [Cartes et plans]

Last update : October 26 1999.
Contact : Anne Bovy (bovy@adre.ucl.ac.be) and Anne Osterrieth (oste@adre.ucl.ac.be)
Responsible : Claire Demain

Figure 2. Example of a design rule.

Once designed, the web pages are passed to the site manager who put them on line and insert them in the local hierarchy, thus leading to several link updates. These pages are therefore made accessible to any visitor with a browser.

According to evaluation needs and on a regular basis (for instance,, each week or each month), the site manager, the local or global web master are performing an evaluation of the current VLWS status. This evaluation covers many facets : information consistency, compliance with style guide, respect of design rules, legibility, absence of broken links, verification of recently published pages, checklist of guidelines,… According to the results, they write a list of found problems, usability errors, and comments (fig. 3) to be sent by electronic mail with a warning to the document author by a webmaster having a strong position. If no problem is detected, nothing is sent.

```
14.4.99
To: NYNS CHARLES-HENRI <nyns@bse.ucl.ac.be>
From: Philippe Degand <Degand@sri.ucl.ac.be>
Subject: Corriger <http://www.bse.ucl.ac.be/index2.html>

* Au bas de la page sous rubrique, le message "Depuis le 6 août 1998, cette
  page a été consultée [an error occurred while processing this directive]
  fois" devrait être corrigé.
* L'appel à <http://www.cdess.org/rcompil.htm> (CDESS) n'aboutit pas.
* Le nom d'un responsable devrait être mentionné au bas de certaines pages
  satellites, de même qu'une date de création ou de mise à jour.
* Enfin, certaines balises "NAME=" ont disparu de la page, alors qu'on y
  fait référence au début par
          <LI><A HREF="#train">en train</A></LI>
          <LI><A HREF="#voiture">en voiture</A></LI>
          <LI><A HREF="#avion">en avion</A></LI>
           <LI><A HREF="#bxl">&agrave; partir de l'UCL-Bruxelles</A></LI>
  La page étant relativement courte, je suggère de
```

```
simplement enlever ces
  pointeurs.
```

Figure 3. Example of a message sent by a webmaster.

In parallel, they maintain files recording the position of the VLWS that have been submitted to the evaluation and their results. When such a warning is received or when new information should be added, the document author re-edits or updates web pages of concern. Updated web pages are sent back to the site manager while an update reply is sent to their site manager and to the webmaster. This reply is typically a message stating what have been updated, what problems have been fixed, what usability errors have been solved. The evaluation log files are updated after verification.

This manual approach cause several shortcomings on a VLWS :

- Due to the size, the complexity and the update frequency, site managers and webmasters are often overwhelmed : they cannot evaluate everything on time, they leave some parts unevaluated, they cannot keep track of all performed evaluations, as requested they are more akin to devote more time to put pages on line than to check their usability.
- The quality of web pages basically relies on the document author background, experience and sake for usability. When bad pages are authored, they go on line before any evaluation can take place. The evaluation may come a long time after.
- Site managers and webmasters do not have the time and the resources to make usable every pages submitted by a document author. For instance, a site manager can receive as much as ten on line documents per day, which is more or less 50 web pages to manage.
- The writing of the list of problems, usability errors, comments, and their sending by e-mail requires too much time for such a repetitive task.
- Warnings and lists sent to document authors are infrequently and partially addressed. For example, some statistics for our university showed that only 40% of document authors provided a reply for the first month and 60% for the second month (table 1).
- Once problems are fixed, usability errors are solved, site managers and webmasters still need to verify the updated pages before updating their evaluation log files. This process is highly iterative (for example, up to 4 or 5 loops before final acceptance).
- Document authors have little or no knowledge on how to apply and check web design guidelines. Moreover, they are not especially aware of recently released guidelines. In particular, they are rarely aware of accessibility guidelines required for all kinds of users although they recognize that these concerns should be supported. For instance, on-line courses should be made highly accessible for distance learning purposes.

	First month	Second month	Mean
Reply with a complete correction	20 %	45%	33 %
Reply with a partial correction	20 %	15 %	17 %
Acknowledge but no correction	40 %	30 %	35 %
No acknowledge and no correction	20 %	10 %	15 %

Rate positive reply/no reply	40/60	60/40	50/50

Table 1. Reply statistics.

These two last shortcomings motivate the need for integrating Universal Design in the global approach [Richarson96, Story98].

3. THE NEED FOR INTEGRATING UNIVERSAL DESIGN

Learning requires complex interactions of the recognition, strategic, and affective systems, and no two brains function in exactly the same way. These are the three main dimensions of Universal Design for Learning [Cooper99]. While everyone's brain functions take place in roughly the same areas and work together in roughly the same way, PET scans show that each individual has his or her own activity "signature." Each of us has a different functional allocation of cortex. Some people have larger regions devoted to recognizing patterns, generating strategies, or focusing on particular priorities and these differences seem to be reflected in different configurations of learning style, relative strengths and weaknesses, and varying "kinds" of intelligence. Thinking about individual differences in light of the three brain systems can help us understand the ways in which curriculum must be flexible to reach all learners. Multiple representations of content can adjust to the recognition systems of different learners; multiple options for expression and control can adjust to the strategic and motor systems of different learners; multiple options for engagement can adjust to the affective systems of different learners [Cooper99].

3.1 Multiple Means of Representation

No single representation of information is ideal, or even accessible, to all learners. Some students thrive in lectures; others obtain information effectively from text, while still others learn best through visual media such as diagrams, illustrations, charts, or video. These learning differences reflect variations in neurology, background experiences, and constitution and are manifested along a continuum from slight preferences to profound necessities. For example, one student with a proclivity for art may find an image more comprehensible than a verbal description of an idea; another who is deaf will be shut out completely if only a verbal description is provided. Universally designed materials accommodate this diversity through alternative representations of key information. Students with different preferences and needs can either select the representational medium most suitable for them, or gather information from a variety of representational media simultaneously. Unlike the printed page, computers provide the opportunity to present information in multiple media and to provide settings that permit selecting among the offerings. Additionally, computers can often transform information into a medium most appropriate for the user. However, it is not always a straightforward matter to do so. In some cases a direct translation is possible, as in text-to-speech or spoken dialogue to written caption. In other cases, interpretation is necessary, as in image description or text version of a sound effect. Some content cannot truly cross media in a way that most people would agree on: a poem or music, for example. It is essential, therefore, when providing multiple representations, to consider the purpose of the activity, and the nature of the learners themselves [Cooper99].

3.2 Multiple Means of Expression

Just as no single mode of presentation suits all learners, neither does any single mode of expression. The dominant mode for expressing ideas and demonstrating learning has long been text on the printed page. Work in multiple intelligences [Gardner, 1983] and school reform supports the notion that more options, including artwork, photography, drama, music, animation, and video, open doors for a greater number of students to successfully communicate ideas, knowledge gained, and talents. These ideas apply to students with particular skills and proclivities as well as to students with disabilities that prevent them from using certain media effectively or at all. Universally designed materials offer multiple options for expression and control. Persons with particular preferences or learning needs can find media, supports, and options that enable them to demonstrate their knowledge in the way that is most effective for them [Cooper99].

3.3 Multiple Means of Engagement

Reaching to users' enthusiasm and interests is critically important. The third principle of Universal Design proposes that media should support varied skill levels, preferences, and interests by providing flexible options. For any given user, there must be content that is interesting and provides a clear purpose. Digital materials and electronic networks have the potential to provide the flexibility, and developers, researchers, and educators will have to ensure that sound pedagogy guides the development of new digital curricula [Cooper99].

4. THE BOBBY TOOL

Bobby™ is a computer-based tool that supports Universal Design of a web site. The notion of a universally designed Web challenges society to think about plurality—to consider all individuals, regardless of age, ability, race, or economic or cultural background—when developing new technologies. Yet at this time, though the Web has much potential for broad inclusion, it often excludes some people from participating in much the same way that a staircase prevents a person in a wheelchair from going in a building's door [Cooper99].

4.1 Universal Design Principles for the Web

The World Wide Web is a potentially rich learning environment. The notion of a universally designed Web challenges society to think about plurality—to consider all individuals, regardless of age, ability, race, or economic or cultural background—when developing new technologies. Yet at this time, though the Web has much potential for broad inclusion, it often excludes some people from participating in much the same way that a staircase prevents a person from going in a building's door.

The technology now exists to support inclusion of many different types of people in ways that were previously unconsidered, yet that technology is not always used to its maximum benefit. For individuals with visual disabilities, for example, the Web's highly graphical environment poses serious problems. Even with a screen reader, a tool used by individuals with visual impairments to translate written text into spoken text [Gappa97, Cooper99], web pages can still be inaccessible when screen readers cannot navigate text in columns or recognize images. For individuals who are deaf or hard of hearing, multimedia and audio elements of Web pages are inaccessible without such accommodations as captioning or text descriptions.

In April 1997, the W3C's establishment launched the Web Accessibility Initiative (WAI) to lead the Web to its full potential by promoting a high degree of usability for people with disabilities [WAI98]. In coordination with other organizations worldwide, the WAI is pursuing accessibility through development of technology, guidelines, tools, education and outreach, and through research and development.

An important piece of the WAI's work has been the development of a document called the Web Content Accessibility Guidelines [WAI98] which brings together all of the previous efforts in this area and provides many new ideas. Within this larger international movement, CAST's tool Bobby has identified a critical need to provide practical support to Web developers in implementing the guidelines [Cooper99].

4.2 Supporting the Authoring of Universally Designed Web Sites

Applying the principles of Universal Design to a web site requires awareness of and commitment to the issues. Equally importantly, it requires enough applied understanding of these issues to create effective universally designed web sites. That is, an author must know the design principles that make a web site universally designed, and the author must know technically how to realize those principles on the web site. To help bring this awareness about, CAST launched Bobby in August, 1996. Bobby is a free interactive tool offered on CAST's web site that analyzes an HTML page with respect to the WAI's Web Content Guidelines, and translates them into instructions for improving its accessibility. After typing in a URL, Bobby delivers a full report within seconds. This report optionally includes the original page, with "Bobby-hat" icons (Figure 4) that visually show the location of errors.

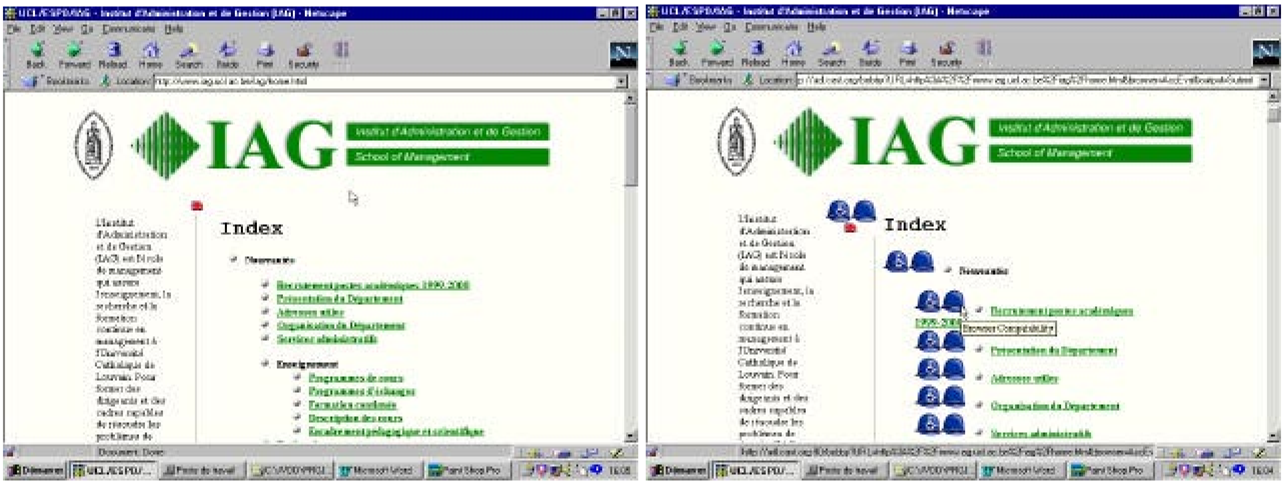


Figure 4. A page evaluated by Bobby. Left: the original page. Right: the page with visual notification of accessibility errors. Clicking on the "hat" provides a more extensive description of the error.

Bobby then explains the factors that limit the site's use and recommends ways to fix those problems. In the report, the factors are presented as a list of error types (fig. 5). For each type, the parts of the page on which it is found is indicated, this time by showing the HTML source. An extended explanation of the cause of the error and means of repairing it is available by clicking on the error title. The errors are organized by three levels of priority—Priority 1 issues are the most important to address for accessibility. Within the priority levels, the report is also grouped into items that it can evaluate automatically, and descriptions of items that require human judgment to determine an appropriate response. While any web page will require an amount of subjective determination, Bobby is able to address many of the most numerous access issues.

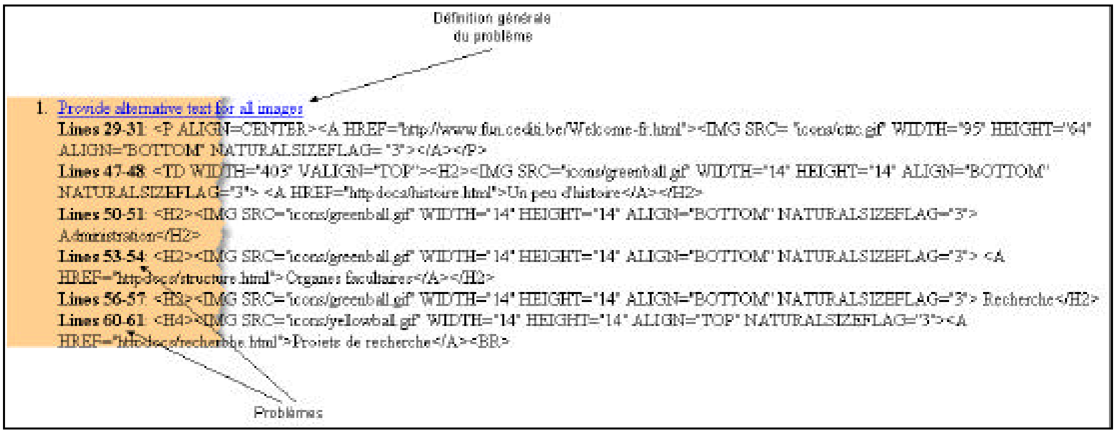


Figure 5. Part of a usability evaluation report.

Now in version 3.1, Bobby has been continually enhanced to provide better support for the guidelines. Many of its recommendations are for alternate representations of media, such as text alternatives and extended descriptions for images; others help authors avoid problems encountered by persons using access aids or non-standard browsers. Bobby can test most of these guidelines. In some cases the test involves detecting the presence or absence of certain features e.g., text alternatives that are included with specific HTML constructs like the ALT or `LONGDESC` attribute of media elements. In other cases, Bobby examines the way certain elements are used, such as color, size, or hierarchical organization.

Bobby is designed to be an educational tool that teaches Web designers about Web accessibility. As Web designers use Bobby, they not only learn how to address problems within their own site, they also learn skills that they can apply to site design in the future. Bobby offers concrete design suggestions and is linked to other sites that discuss access issues. The more one uses Bobby, the less likely one is to need it in the future, as accessibility issues and their solutions become integrated into one's Web design at the outset [Cooper99].

In order to serve as an effective model of accessibility and good interface design, Bobby employs the latest technological innovations in its own design. Bobby is now written in 100% Java, and has two forms: the online server, and a downloadable version that uses the same page evaluation code and offers both a graphical and a command-line interface. Since it is written in Java, this version can run on many different hardware platforms. Bobby uses Java's most current accessibility features, which allows the program itself to be accessible to users with disabilities [Glinert92]. Many access aids are built into the interface, and it has the requisite code to allow third party access aids to communicate with it effectively.

The accessibility report will consist of at most seven sections (some sections are not displayed if irrelevant): Priority 1 accessibility errors, Priority 2 errors, Priority 3 errors, browser compatibility, and download time. The online Bobby will redisplay the web page that you asked it to analyze appending an accessibility report to the bottom of the page if the "Text only output" option is not checked [CAST97,Cooper99]:

- The **Priority 1 accessibility errors** section lists problems that seriously affect the page's usability by people with disabilities. A Bobby Approved rating can only be granted to a site in which none of the pages have accessibility errors. Clicking on any of the problems that Bobby reports will produce a more detailed description of how to fix the problem. In addition to items that Bobby can examine automatically, a number of items that require manual examination are presented here. You must be able to answer affirmatively to these questions. The responses to these questions affect your Bobby Approval rating since they are important to ensure your site is accessible in accordance with Priority 1 WAI guidelines. Bobby Approved status is equivalent to Conformance Level A for the Web Content Guidelines.
- **Priority 2 access errors** are access problems which you should try to fix. Although not as vital as Priority 1 access errors, the items in this section are considered important for access. There are items presented here as well that require manual examination. If you can pass all items in this section, your page meets Conformance Level AA for the Web Content Guidelines. This is the preferred minimum conformance level for an accessible site, even though it is not considered part of Bobby Approved.
- **Priority 3 access errors** are third-tier access problems which you should also consider. There are items presented here as well that require manual examination. If you can pass all items in this section, your page meets Conformance Level AAA for the Web Content Guidelines.
- The **browser compatibility** section lists those HTML elements and element attributes that are used on the page which are not valid for particular browsers.
- The **download time** section provides a summary of how long the web page and images would take to download on a slow modem line (assuming the server is not too busy).

4.3 Tool support for WAI guidelines

Bobby 3.1 is an improved implementation of the working draft of the Wide Web Consortium's W3C's Web Access Initiative (WAI) Page Authoring Guidelines [WAI98] as well as reflecting the Page Authoring Guideline Working Group's latest revisions to them. There are, however, some aspects of page design that are important to accessibility but can not be tested automatically by Bobby. Table 2 lists some excerpts of the current WAI guidelines, and the type of support that Bobby provides.

Table 2. WAI guidelines as supported by Bobby (excerpts from http://www.cast.org/bobby/faq.html).

Guideline 1. Provide equivalent alternatives to auditory and visual content.

Tech.#	Guideline	WAI Rating	Bobby Support
1.1	Provide alternative text for all images	p1	full
1.1	Provide alternative text for each APPLET	p1	full
1.1	Provide alternative content for each OBJECT that conveys information	p1	full
1.1	Provide alternative text for all buttons in forms	p1	full
1.1	Use separate buttons or images with ALT text for form controls	p1	full
1.1	ALT text too long, consider providing a separate description	p1	manual
1.1	If any of the images on this page convey important information beyond what is in each image's alternative text, add descriptive (D) links	p1	manual
1.1	If any of the images on this page convey important information beyond what is in each image's alternative text, add a `LONGDESC` attribute	p1	manual
1.1	Do all audio files have transcripts?	p1	manual
1.1	Have you provided audio descriptions for short visuals like animated GIFs?	p1	manual
1.1	Did you provide a synchronized textual transcript for the audio associated with this video?	p1	manual
1.1	Avoid ASCII art if it is important information. Replace it with an image and alternative text	p1	manual
1.2	Provide alternative text for all image map hot-spots	p1	full
1.2	Is this image button being used as a server-side image map?	p1	partial
1.2	Client-side image map contains a link not presented elsewhere on the page	p2	partial
1.2	Provide redundant text links for each active region of a server-side image map.	p3	full
1.3	Does all video information have both a description and a synchronized caption?	p1	manual
1.4	Have you provided visual notification and transcripts of sounds that are played automatically?	p1	partial

Guideline 2. Don't rely on color alone.

Tech.#	Guideline	WAI Rating	Bobby Support
2.2	Use foreground and background color combinations that provide sufficient contrast	p2	partial
2.2	Make sure that document structure is supported by the proper use of structural elements	p2	manual

Guideline 3. Use markup and style sheets properly.

Tech.#	Guideline	WAI Rating	Bobby Support
3.1	Style sheets should be used to control layout and presentation wherever possible	p2	partial
3.1	Where it's possible to mark up content (for example mathematical equations) instead of using images, use a markup language (such as MathML).	p2	manual
3.2	Make sure that headings are nested properly	p2	partial
3.3	Only use list elements for actual lists, not formatting	p2	partial
3.4	Mark up quotations with the Q and BLOCKQUOTE elements	p2	manual
3.7	Use relative sizing and positioning (% values) rather than absolute (pixels)	p2	partial

Guideline 4. Clarify natural language usage.

Tech.#	Guideline	WAI Rating	Bobby Support
4.2	Use the ABBR and ACRONYM elements to denote and expand abbreviations and acronyms.	p3	partial
4.3	Identify the language of the text, and any changes in the language	p3	partial
4.3	If a resource is served in various formats or languages, use content negotiation to determine the format or language preferred by the user.	p1	manual

Guideline 5. Create tables that transform gracefully.

Tech.#	Guideline	WAI Rating	Bobby Support
5.2	If this table contains data in rows and columns (i.e. a spreadsheet), have you identified headers for the table rows and columns?	p2	partial
5.3	If possible, avoid using tables to format text documents in columns.	p2	partial
5.5	If this table is used to display data in rows and columns (i.e. a spreadsheet), have you provided a summary of the table.	p3	partial
5.6	Provide abbreviations for lengthy row or column labels.	p3	partial

Guideline 6. Ensure that pages featuring new technologies transform gracefully.

Tech.#	Guideline	WAI Rating	Bobby Support
6.1	Ensure that pages are readable and usable without frames	p1	full
6.1	Make sure that style sheets transform gracefully	p1	manual
6.2	Ensure that descriptions of dynamic content are updated with changes in content.	p1	manual
6.3	Provide alternative content for each SCRIPT that conveys important information or function	p1	manual
6.3	Is there a more accessible way to implement this applet?	p1	manual
6.4	Make sure event handlers are device independent for programmatic objects.	p2	manual
6.5	Ensure that dynamic content is accessible or provide an alternate presentation or page.	p2	manual

Guideline 7. Ensure user control of time-sensitive content changes.

Tech.#	Guideline	WAI Rating	Bobby Support
7.2	Avoid blinking or scrolling text created with the MARQUEE element	p2	full
7.2	Avoid blinking or scrolling text created with the BLINK element	p2	full
7.3	Did you avoid using movement where possible?	p2	partial
7.3	Did you provide a mechanism to allow users to freeze movement or updating in applets and scripts	p2	manual
7.4	Is there an alternative page where "auto-refreshing" is only done on the users request (manual refreshing only)?	p2	partial

Guideline 8. Ensure direct accessibility of embedded user interfaces.
Guideline 9. Design for device-independence.
Guideline 10. Use interim solutions.
Guideline 11. Use W3C technologies and guidelines.
Guideline 12. Provide context and orientation information.
Guideline 13. Provide clear navigation mechanisms.
Guideline 14. Ensure that documents are clear and simple.

Level \ Support	Manual	Partial	Full	Total
1	16	2	8	26
2	14	18	2	34
3	7	6	1	14
Total	37	26	11	74

Table 3. Level of support for WAI guidelines by Bobby.

The big advantage with web sites is that their HTML code can be downloaded and examined remotely, which is not the case for traditional interactive applications. For these applications, it is reported [Farenc96, 97] that 44% of guidelines relating to interaction objects of a user interface can be evaluated in an automated way. The rest either cannot be automated or can only be processed if more than the resource files are accessible. It is therefore expected that the automated evaluation of web design guidelines will go beyond this barrier thank to the code accessibility. Table 3 shows the different levels of support provided by Bobby for WAI guidelines. If we sum up the partial and full support, we can reach the percentage of 50% of guidelines automatically processed (fig. 6), which is only a little bit beyond the 44% barrier.

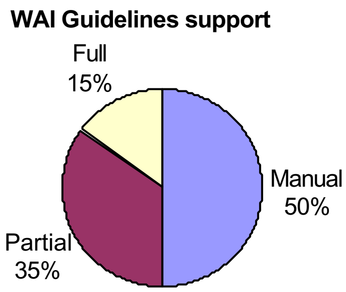


Figure 6. Repartition of support level for WAI guidelines by Bobby.

5. THE EXTENDED BOBBY ENVIRONMENT

In order to overcome the shortcomings discussed in section 2 and to integrate Universal Design as described in section 3 into a global approach for managing VLWSs and in order to support it by Bobby (section 4), we choose to extend the Bobby environment with the following principles and tools (fig. 7) :

- **Evaluation on demand** : up to now, Bobby only evaluates the WAI accessibility guidelines while it could be equally important to see Bobby evaluate when possible other sets of guidelines or any combination of guidelines extracted from several guidelines base. Evaluation on demand promotes the definition of any combination of guidelines extracted from several sources (and potentially conflicting or inconsistent) and the evaluation of this combination. We therefore propose to extend Bobby with Application Programming Interfaces which are able to communicate the

HTML code extracted by the Bobby parser to the different combinations of guidelines.

- **Evaluation of custom guidelines** : many companies have developed their own corporate environment style guide containing specific guidelines that may not appear in existing sets of guidelines. Moreover, some guidelines can come from design rules decided by the company. In order to support the evaluation of custom guidelines, we propose a guidelines based editor with which a designer is able to graphically specify guidelines relating to graphical aspects. Guidelines that cannot be expressed graphically should therefore be coded separately, for instance as functions developed in an appropriate programming language. These custom guidelines should be easily incorporated in any combination of guidelines to be evaluated.
- **Computer-aided evaluation of guidelines** : as seen in fig. 6, almost 50% of WAI guidelines can be evaluated automatically by Bobby. It is expected that most of the guidelines that can be processed by an automata are supported by a software that evaluate any web page as automatically as possible. On the other hand, human control over the evaluation process is also a key feature so that the evaluation can be launched in a completely automated way or with human supervision during the evaluation process.
- **Definition of evaluation tasks** : since evaluation tasks are repetitive and can partially be automated, it would be helpful to have an evaluation task editor enabling an evaluator to define parameters of an evaluation task to be performed by Extended Bobby. Such parameters could include :
  - the starting URL, e.g., http://www.qant.ucl.ac.be
  - the maximum link level up to which pages should be evaluated, e.g, up to level 3
  - the need for recursive evaluation, e.g., with all subdirectories
  - the reference to one or many combinations of guidelines that need to be evaluated, e.g., guidelines 1 through 9 from the WAI, Part 12 of the ISO 9241 standard, guidelines 1 through 25 of a custom guidelines base
  - the severity level with which web pages should be evaluated, e.g., with the most important guidelines only
  - the periodicity of the evaluation, e.g., launch this evaluation task every Friday at 5 p.m.
  - the option of re-checking previously evaluated web pages, e.g., re-launch this evaluation task now after it has been processed two times already
  - the option to generate a site map on the fly, e.g., with site map generation linking bad web pages
  - the options for generating a usability report : here, multiple formats and levels of details should be supported
  - the option for sending a user notification by e-mail to the document responsible person
  - the option of considering or forgetting previous evaluations, e.g. forget previous evaluations of this part since it today contains new pages
  - the option for building a proposal for repairing the bad web pages (see next point)
  - the record of the evaluation results into log files

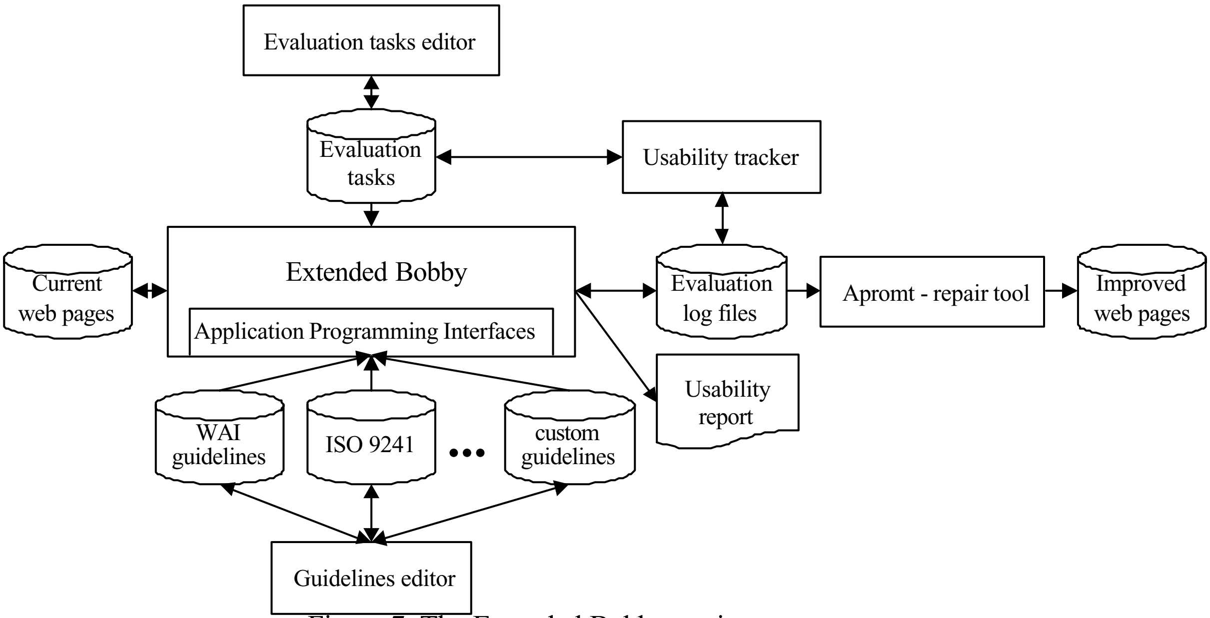


Figure 7. The Extended Bobby environment.

- **Repair support for pages with problems** : the A-Prompt project [Aprompt99] is intended to develop a repair tool that automatically produces a proposal for new HTML code for each page evaluated with problems by Extended Bobby. One or several proposals can be made according to the parameters of the evaluation task. The results of this can be sent with the warning to the document responsible person at the same time.
- **Usability site tracker** : this tool exploits the evaluation tasks defined by the evaluation editor and the evaluation log files produced at evaluation time by Extended Bobby. According to the results, the evaluation frequency or any reply from a document responsible person stating that a new web page has been put on-line, the site tracker should keep track of all detected problems, usability errors and so forth. This feature will guarantee that they will be fixed, solved in a certain amount of time. The main goal of this tool is to release evaluators from repetitively re-evaluating web pages that have been evaluated before, from the management of e-mails with persons (e.g., once updated, a document author can send a predefined message to notify the usability site tracker that a repaired page has replaced an existing page).
-

6. THE ENVISIONED GLOBAL APPROACH

The envisioned global approach for managing VLWSs with the above tools is outlined in fig. 8.

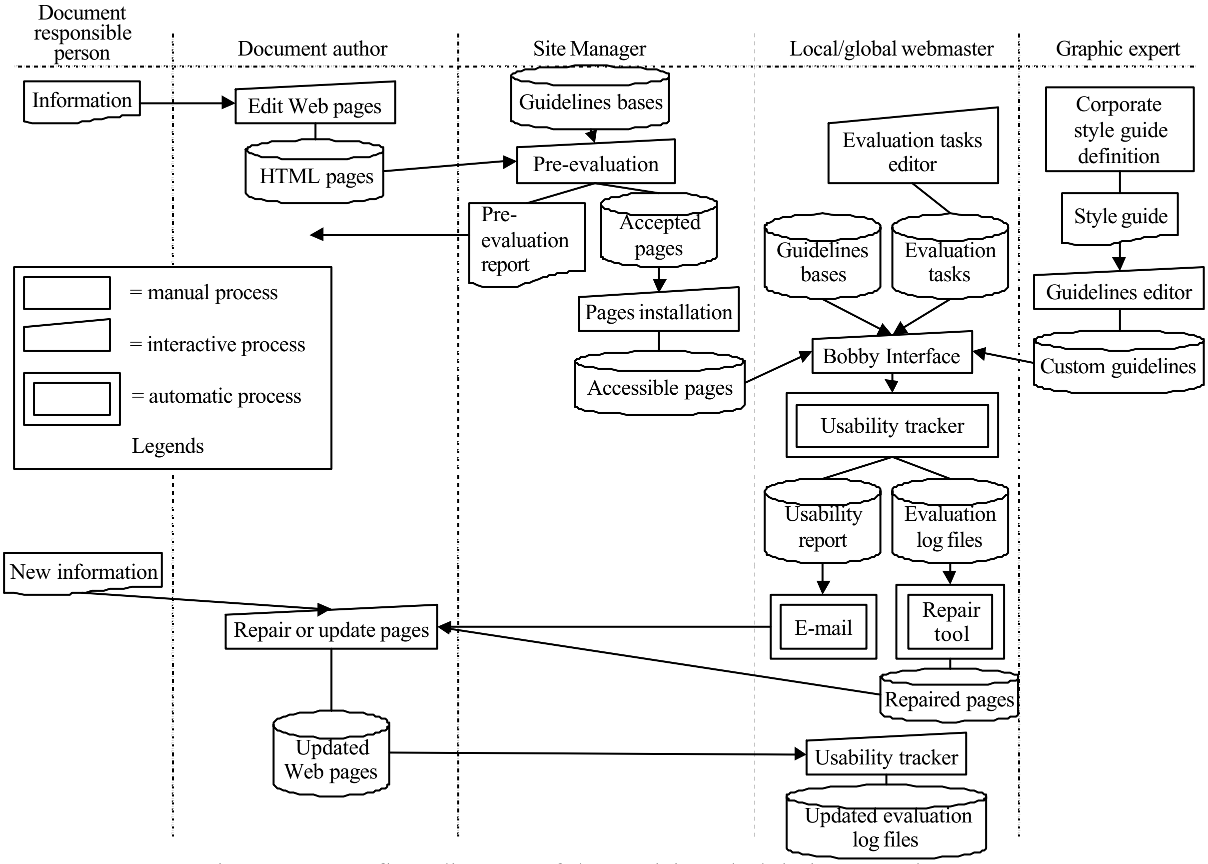


Figure 8. Data flow diagram of the envisioned global approach.

The main differences with respect to fig. 1 are the following :

- In order to prevent document authors to produce web pages with problems, the site manager can perform a pre-evaluation of submitted pages before publishing them on-line. This pre-evaluation can typically consists in a static analysis of each web page, in particular for presentation aspects and individual accessibility issues. Navigation aspects are hard to evaluate off-line. For this purpose, the site manager can define a typical evaluation task of any submitted web page across guidelines contained in guidelines bases. As long as this pre-evaluation is not satisfactory, a pre-evaluation report is sent back to the document author and the web pages remain in the temporary pool.
- The local/global webmaster can define respective evaluation tasks according to their specific needs, for instance the evaluation of some sub-parts of the VLWS with respect to navigation, accessibility, etc. These evaluation tasks can be processed by Extended Bobby either in automatic mode or in computer-aided mode. The usability site tracker is then informed by the evaluation results to record them into evaluation log files. According to parameters, a usability report is produced and sent back and/or proposals for repairing the accused pages.
- If the document author replaces an accused page by an improved one or a repair proposal, s/he can send a predefined message to the usability site tracker to record the modification. The evaluation log files are updated accordingly.
- A graphic expert can independently define the combinations of guidelines that need to be evaluated in any evaluation task. For this purpose, s/he can select subsets of

guidelines from different previously defined source and gather them in a specific guidelines base. Moreover, the guidelines that are not part of standards documents such as style guides, standards, can be defined separately and reused at evaluation time. This is specifically intended to support custom guidelines.

7. CONCLUSION

The data flow outlined in fig. 8 is only a vision for a global approach for managing a VLWS while considering Universal Design and keeping the evaluators' work load to a minimum. We are currently working on the mechanization of guidelines contained in guidelines bases. Out of the multiple formats a guideline can take, it is very likely that the final format will be a programming function for each guideline. To identify a guideline that can be calculated, we are looking at the complexity theory and calculability to see if a guideline can be calculated in the sense of the calculability thery.

It is also very likely that such an approach will raise new types of computational questions, organizational questions such as :

- how does the Extended Bobby deal with a very large number of guidelines?
- what will happen if Extended Bobby reports rule violations in 5,000 pages?
- will the repair tool be able to automatically correct most of them?
- how does Extended Bobby will deal with conflicting guidelines?
- how can the guidelines base be updated and by whom?

REFERENCES

[Access98] Accessibility and universal design: advantages and impact for business, government and developers, *Computer Networks And Isdn Systems*, 30(1-7), April 1998, pp. 759–760. Accessible at http://www.science server.com/journals/01697552/sz986882/98001330.html

[Aprompt99] A-Prompt Accessibility Toolkit Project, Adaptive Technology Resource Center (University of Toronto) and Trade Center (University of Wisconsin), Canada & USA, April 1999. Accessible at http://aprompt.snow.utoronto.ca/

[Bastien95] J.M.C. Bastien, D.L. Scapin, Evaluating a User Interface with Ergonomic Criteria, *International Journal of Human-Computer Interaction,* 7, 1995, pp. 105–121.

[Bastien96] J.M.C. Bastien, D.L. Scapin, C. Leulier, Looking for Usability Problems with the Ergonomic Criteria and with the ISO 9241-10 Dialogue Principles, in *Companion Proc. of CHI'96*, Vancouver, April 13-16, 1996, ACM Press, New York, 1996, pp. 77–78. Accessible at http://www.acm.org/sigchi/chi96/proceedings/intpost/Bastien/jmcb_txt.htm

[Bergman95] E. Bergman, E. Johnson, Towards Accessible Human-Computer Interaction, in "Advances in Human-Computer Interaction", Vol. 5, J. Nielsen (ed.), Ablex Publishing Corporation, Norwood, 1995. Accessible at http://www.sun.com/tech/access/updt.HCI.advance.html

[CAST97] Bobby Version 2.0, HTML Web Analyzer, Center for Applied Special Technology (CAST), National Universal Design Laboratory, Peabody, Massachusetts, 1997. Accessible at http://www.cast.org/bobby.

[Connell97] B.R. Connell, M. Jones, R. Mace, J. Mueller, A. Mullick, E. Ostroff, J. Sanford, E. Steinfeld, M. Story, G. Vanderheiden, The Principles of Universal

Design, Version 2.0, The Center for Universal Design, North Carolina State University, Raleigh, 1 April 1997. Accessible at http://www.design.ncsu.edu/cud/pubs/ udprinciples.html

[Coombs99] N. Coombs, Equal Access to Software and Information (Easi-Web), On-line workshop on Universal Design Principles for Web pages, TLT Group, Rochester University of Technology, Rochester, 1999. Accessible at http://www.nyise.org/whatsnew/easiweb.html

[Cooper99] M. Cooper, Evaluating Accessibility and Usability of Web Pages, Chapter 3 in "Computer-Aided Design of User Interfaces II", *Proc. of 2nd Int. Conference on Computer-Aided Design of User Interfaces CADUI'99*, Louvain-la-Neuve, October 21-23, 1999, J. Vanderdonckt & A.R. Puerta (Eds.), Kluwer Academics, Dordrecht, 1999, pp. 33–42.

[Farenc96] Ch. Farenc, V. Liberati, M.-F. Barthet, Automatic Ergonomic Evaluation : What are the Limits ?, in *Proc. of 2nd Int. Workshop on Computer-Aided Design of User Interfaces CADUI'96*, Namur, 5-7 June 1996, J. Vanderdonckt (ed.), Presses Universitaires de Namur, Namur, 1996, pp. 159–170. Accessible at http://lis. univ-tlse1.fr/~farenc/papers/cadui96-cf.ps

[Farenc97] Ch. Farenc, ERGOVAL : une méthode de structuration des regles ergonomiques permettant l'évaluation automatique d'interfaces graphiques, Ph.D. thesis, Université de Toulouse I, January 1997. Accessible at http://lis.univ-tlse1.fr/~farenc/these.htm

[Gappa97] H. Gappa, R. Opperman, M. Pieper, Certifying Web Accessibility for the Handicapped by ISO 9241 Conformance Testing, in *Proc. of 3rd ERCIM Workshop on "User Interfaces for All"*, Obernai, November 3-4, 1997.

[Gardner83] H. Gardner, *Frames of Mind: the theory of multiple intelligences*, BasicBooks, New York, 1983.

[Glinert92] E.P. Glinert, B.W. York, Computers and People with Disabilities, *Communications of the ACM*, 35(5), 1992, pp 32–35.

[Grose98] E.M. Grose, Ch. Forsythe, J. Ratner (eds.), *Human Factors and Web Development*, Lawrence Erlbaum Associates, Mahwah, 1998. Table of contents accessible at http://www.erlbaum.com/2704.html

[IBM97] IBM Web Design Guidelines, International Business Machines, 1997. Accessible at http://www.ibm. com/IBM/HCI/guidelines/web/web_guidelines.html

[Lowney96] G. Lowney, Design Guidelines to Make Sites Accessible, in *Proc. of 3rd Conf. on Human Factors and the Web*, October 1996. Available at http://www.microsoft.com/usability/webconf/lowney/lowney.htm

[Nevile96] L. Nevile, P. Batchelor, 'Handling' a University's Website, in *Proc. of 3rd Australian Conf. On the World Wide Web AusWeb'97*, 5-9 July 1997. Accessible at http://ausweb.scu.edu.au/proceedings/nevile2/ index.html

[Ratner96] J. Ratner, E.M. Grose, E.M., Ch. Forsythe, Characterization and Assessment of HTML Style Guides, in *Companion Proc. of CHI'96*, Vancouver, April 13-16, 1996, ACM Press, New York, 1996. Accessible at http://www.acm.org/sigchi/chi96/proceedings/intpost/Ratner/ rj_txt.htm

[Richardson96] S.J. Richardson, D.F. Poulson, In Search of Universal Principles for Interface Design, in *Proc. of 1st Int. Conf. on Applied Ergonomics ICAE'96*, Istanbul, 21-24 mai 1996, A.F. Özok & G. Salvendy (eds.), USA Publishing, Istanbul - West Lafayette, 1996, pp. 244-249.

[Scapin90] D.L. Scapin, Organizing Human Factors Knowledge for the Evaluation and Design of User Interfaces, *International Journal of Man-Machine Studies*, 2(3), 1990, pp. 203–229.

[Stephanidis98] C. Stephanidis, G. Salvendy, D. Akoumianakis, N. Bevan, J. Brewer, P.L. Emilianis, A. Galetsas, S. Haataja, I. Iakovidis, J. Jacko, P. Jenkins, A. Karshmer, P. Korn, A. Mareus, H. Murphy, Ch. Stary, G. Vanderheiden, G. Weber, J. Ziegler, Towards an Information Society for All : An International Research & Development Agenda, *International Journal of Human-Computer Interaction*, 10(2), 1998, pp. 107–134.

[Stephanidis99] C. Stephanidis, G. Salvendy, P.L. Emilianis, D. Akoumianakis, Policy Perspectives on "Design for All", in *Proc. of 8th Int. Conf. on Human-Computer Interaction HCI International'99*, Munich, 22-26 August 1999, H.-J. Bullinger & J. Ziegler (eds.), Vol. II, Lawrence Erlbaum, Mahwah, 1999, pp. 868–872.

[Story98] M.F. Story, Maximising Usability: The Principles of Universal Design, *The Assistive Technology Journal*, 10(1), 1998, pp. 4–12.

[Usable98] Usable Web. Accessible at http://www.usableweb.com, http://usableweb.com/item/chi.html

[UseIt98] Use It. Accessible at http://www.useit.com

[WAI98] WAI Accessibility Guidelines – Page Authoring, Web Accessibility Initiative, World Wide Web Consortium, Geneva, 1998. Accessible at http://www.w3.org/TR/1998/WD-WAI-PAGEAUTH-0203

[Vanderdonckt99] J. Vanderdonckt, *Development Milestones towards a Tool for Working with Guidelines*, Interacting with Computers, Vol. 12, N°2, 1999.

[Washington99] Accessible Web Design, or, Effective Web Page Development Using Universal Design, Washington University, Washington, 1999. Accesible at http://weber.u.washington.edu/~doit/UA/DEMO/index.html

[Yale97] Yale C/AIM Web Style Guide, Yale-New Haven Medical Center, January 1997. Accessible at http:// info.med.yale.edu/caim/manual/contents.html

Access to Computer-Assisted Learning Environments for Severely Handicapped Children by Semantic Level Adaptations

Henrike Gappa and Stefanie Mermet

GMD-German National Research Center for Information Technology
D-53731 Sankt Augustin (FRG)
Tel.: +49 2241 142793/2044
Fax: + 49 2241 142065
E-mail: henrike.gappa@gmd.de, stefanie.mermet@gmd.de

Abstract: Within the project CATI (Computer-Aided Therapeutic Intervention) new potentials and limitations of computer-supported multimedia applications for the therapeutic advancement of severely handicapped children are investigated. In this context, we are conducting a case study involving a 12-year old physically handicapped child with behaviour typical of autism. A memory game featuring various individualised semantic adaptations was developed for the training of metacognitive skills. The design of the training adopts procedures used for cognitive behaviour modification. A significant other defined as a person with high social and rewarding power is used for instructional purposes in the computer-assisted learning environment. It is to investigate how the readiness to attend and the achievement motivation can be enhanced by the implementation of such a person. First results show that the motivation to stay on task is developing and the goal of achieving problem solving techniques appears to be attainable.

The potentials of information technology for the compensation of functional handicaps became already apparent in various research projects (PIEPER et al. 1998, PIEPER et al. 1999). However gaining access to computer-assisted learning environments by adaptations on the semantic level of software systems is still to be investigated for children with special needs so far unable to use this educational tool.

Within the project CATI (Computer-Aided Therapeutic Intervention) a case study is conducted involving a 12-year old severely handicapped child. He is physically impaired and exhibits behaviour as described for autism. So far, he is unable to use computer-assisted training materials due to attention deficits and a lack of problem solving techniques. Therefore, a computer-assisted memory game for the training of metacognitive skills was developed featuring various individualised adaptations on the semantic level of the learning environment.
The adaptations include the implementation of a socially significant other for instructional as well as interventional purposes referring to a person with high social

power. Interventional strategies are derived from procedures used for cognitive modelling and are employed according to the results of the constant evaluation of the problem solving behaviour of the child. In order to achieve a better understanding for the background of the design of the computer-assisted learning environment, the handicap of the child concerned will be described in short.

1. The CATI Case

The child participating in this case study is suffering from cerebral palsy, a motor impairment due to brain damage. In this case, the handicap affects all extremities, so walk, the use of arms, hands and fingers is functionally impaired but possible without prosthetic measures. As a result of brain damage, perceptional deficits occur rather often (BATSHAW et al. 1992) and seem to be present also in this case.

Perceptional deficits influence all information processing procedures from the stimulus input to the integration of the perceived stimuli. Attentional deficits due to inhibited selective processes during the attention focussing phase as well as inappropriate reactions are the consequence. Relevant stimuli are often not detected and deficient variability in the reactive repertoire causes the rigid holding on reactions once learned.

The child concerned in this case study strongly prefers information input by the acoustic channel. He likes to listen to music and also likes to play on a piano. He shows a remarkable skill in memorizing text and melody of songs for a long time as opposed to rather poor memory skills in other academic areas. So far, he is not able to read or write and does not understand numbers. He is attending a public junior high school, which mainstreams special needs children in a model project for the first time.

The primary physical handicap of this child is accompanied by secondary handicaps like behaviour typically described for children suffering from autism. This concerns mainly his language abilities and behavioural disturbances. He repeats seemingly senseless and incoherent words and sentences (immediate echolalia) and asks stereotypically questions may be with the goal of finding back into a conversation.

Beyond disabled speech and language abilities, behaviour disturbances as described for autism occur. Permanent repetition of seemingly meaningless behaviour called perseverating behaviour pattern prevails for instance when using a computer-mouse. He permanently pushes the mousebutton without any goal-orientation.

2. Technological Requirements on the Input Device

Due to the severe impairments of the child as described above, it is impossible for him to use the standard equipment for operating a computer. An alternative input device needed to be searched and tested for. The results showed, that a touchscreen as input device seems to be the most appropriate, since his fine and gross motor control allows for isolating one finger and touching a given area on a computer screen (see figure 1).

Figure 1

During the testing, we payed very much attention to what input could be carried out independently, that means nobody needed to serve as a facilitator for operating the computer. This issue appeared to us very important, because from a motivational point of view independency of performance is crucial for the achievement of self-esteem. The outcome of an action needs to be attributed to one's own abilities in order to trigger off reinforcing effects (HECKHAUSEN 1987). Furthermore, the discussion about authenticity of input facilitated by somebody else could be avoided (CUMMINS et al. 1992).

3. Methodological Basis of the Developed Learning Environment

The shift from understanding learning disability as an information processing problem rather than a structural deficit evoked the development and evaluation of training programs for cognitive behaviour modification (SCRUGGS et al. 1993, LAUTH et al. 1997). The training of metacognitive skills is part of these training programs, since the knowledge about cognitive processes is crucial to the success of the intervention. Metacognitive skills in this sense include the proper comprehension of a given problem, the mental representaion of the problem, the conscious selection of a suitable problem solving strategy and finally the evaluation of the outcome of the problem solving process (DAVIDSON et al. 1994).
The child considered in this case study is most obviously affected by an attention deficit influencing the (meta)cognitve processes as described. He is unable to select relevant stimuli and his problem solving behaviour is guided by an impulsive cognitive style. In addition, he does not evaluate the result of his reaction. Therefore, the development of a computer-assisted training aimed at the advancement of cognitive control in accordance with the cognitive training programs as mentioned above appeared to be most appropriate. In the first step, the selection of relevant stimuli as well as a systematic and reflective strategy for solving a given task were determined as the goal of accomplishment.

The didactic strategy to convey problem solving strategies in the developed computer-assisted learning environment incorporates methods derived from procedures used for scaffolding (TINZMANN et al. 1990). Scaffolding is an instructional technique whereby the teacher models the desired learning strategy. Cognitive as well as behavioural aspects of the problem solving process are verbalized and demonstrated by this model. Research shows, that this and similar working methods are successful in conveying problem solving behaviour and became established training programs for cognitive behaviour modification in special education (BROWN et al. 1990, HASSELHORN et al. 1990). However, the success of these methods depend on the ability of the observer to imitate somebody. Due to the autistic behaviour pattern of the child concerned in this case study, it was not expected that he would imitate a model. As described in chapter 4, the procedures of the scaffolding strategy, though, were incorporated in the training and formulated as questions or instructions.

Besides considering the didactic strategy to achieve the learning goal, it was very important to ensure that attention as the precondition for learning can be evoked by the computer-assisted learning environment. So far, the child concerned strongly depends on somebody else for attention focussing. He always needs to have somebody at his disposal. Therefore, a very important goal of the computer-asssisted learning environment was to enable him to work independently as much as possible.

In this context, it is important to state, that the goal of the study is by no means to withdraw personal care und support by other people. It is rather aimed at conveying learning competence, so the child involved will be able to work independently as it is required for attending a regular school and working with standard educational software.

“Human support“ was implemented in the learning environment by the employment of a socially significant person mainly for instructional purposes. As it is known from social psychology, ”significant others“ gain social power because their opinion about and reflected to the person concerned, influences his or her self-concept (BACKMANN 1974). Further, the degree of influence a person can exert on somebody else is defined by the attributed degree of social power (BACKMANN 1974). Social power in this sense has many facets. The person can have the power to reward, coerce, provide emotional support and the like. In this case study a significant person with high social power through attribution of sympathy and competence by the child involved was expected to be most successful. A person with these attributions is also expected to have the most efficient rewarding power necessary for reinforcement. The investigation will show, whether an individually significant person can retain social power and gain attention even though he or she is present only virtually.

Due to many failures experienced with learning, this area is associated with ambigous feelings for the child participating in this case study. Therefore, a comic figure was implemented for prompting the correct responses in case the intervention of the significant other person was not sufficient. This way, it could be avoided that the relation between child and the implemented significant other would be burdened with feelings of failure which often leads to withdrawal as the child tends to.

For the presentation of visual information, criteria valid for the instruction of children affected by a so- called attention deficit disorder need to be considered (NADDA 1999). In order to avoid stimulus overflow, colouring, for instance, is used carefully for the facilitation of selective perceptional procedures. The motives of the picture cards of the memory game are chosen according to the personal preferences of the child concerned in order to employ the motivational aspect of individually meaningful information. Training material like completion of samples and the like as it is used in common computer-assisted attention training programs (see SODIS 1999) would not allow for the consideration of these issues, so the need for designing new picture cards became apparent. Since the child likes animals very much, we designed 27 memory cards with common animals and typical sounds they produce.

Since children affected by attention deficits lose the attentional focus while waiting for response to their actions, they are not able to deal with delayed responses. Thus, it is crucial to the success of the training that the learning environment provides immediate feedback. The possibility of a computer to give always immediate feedback is an already well known strength of computer-assisted learning as opposed to group learning.

4. Description of the Memory Game

In the following, a short description of the developed memory game is presented in order to illustrate how the methodological considerations as described are realized in the game. So far, there are two modes of the game available. In the first mode up to 8 picture cards are shown on a computer screen presenting only one pair of animals (see figure 2). All the other picture cards show different animals. This is a discrimination task geared at conveying the concept of ”the same“, since this is the underlying concept of a memory game.

Figure 2

The instructions are all given acoustically as the child involved is unable to read so far. He is addressed very often by his name in order to gain attention. Each step of instruction or intervention is formulated in different ways expressing the same aspect, because he easily learns things by heart preventing him from reacting to the verbal input. In case the response was correct, a short sound-track including, for example, how this animal “calls“ for other group members is presented.

During the whole problem solving process, the developed software evaluates all given input and intervenes, based on an accordingly derived user model, by verbalizing the required behavioural and/or cognitive aspect, e.g. requests or questions like: “Take a close look…„ (behavioural aspect), “Compare all animal pictures one by one. Which pictures show the same animal?“ (cognitive strategy), “Look for two picture cards, that…“ (attention (re)focussing). Due to the reduced attention span, instructions need to be very brief and are rather subdivided into several steps.

If verbal intervention does not lead to a successful response, a comic figure (a magician called “Mirko“) is shown on the computer screen prompting the correct reponse by “sitting next“ to the picture card looked for. It is pointed out to the child, that Mirko will show him the animals, that look the same. In case the child is still not able to find the correct answer, the solution is presented and a new task appears. After a certain amount of correct responses, a reward is implemented by a comic figure looking like a little boy dancing on stage according to the favourite songs of the child.

The second mode of the memory game presents up to 4 pairs of picture cards with animals (see figure 3). At first all animals on the picture cards are shown. After about 5 seconds, all picture cards are covered and the child is asked to remind where he has seen animals that look the same. The intervention and reward strategy was selected according to the first mode of the game.

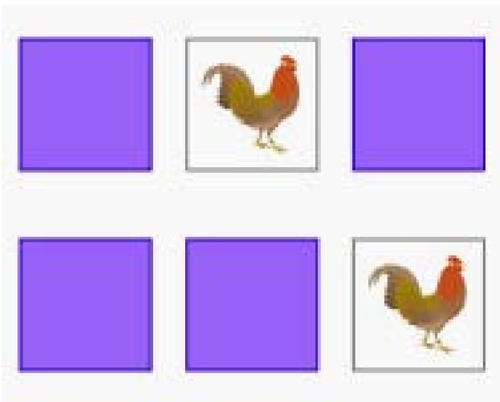


Figure3

5. First results

The computer-assisted learning environment has recently been implemented. Each session is recorded by means of a short questionnaire rating achievement motivation, degree of independency while working and the like. Furthermore, a log file recording all clicks on the touchscreen in accordance with the instructional/interventional step was installed. First evaluations of the questionnaire showed, that the child shows good achievement motivation and pays attention to the instructions presented by the socially significant person. He shows very much interest in working with this person via the computer, stays on task and reacts to initiated interventions. Furthermore, he enjoys listening to the animal sounds and watching the comic figure while his favourite songs are played. The touchscreen as input device has proven to be appropriate. After some training sessions, the child concerned was able to operate the computer independently.

Thus, the first step is taken as the child concerned has gained access to a computer-assisted learning environment which appeared to be impossible so far. This is already a success. However, all the observed effects might be due to novelty. In order to evaluate thoroughly the methodological concepts leading to the individualised adaptations on the semantic level of the developed system as described, a working period of at least three months is regarded as necessary. After that period of time, data will also be available allowing for an evaluation of the learning process while working with the learning environment.

Since the presented investigation is conducted as a case study, results are not to generalize. However, the investigation will give evidence about an innovative approach of designing learning software for children with special needs unable to benefit from standard educational software. The approach uses new multimedia technology for implementing a socially significant person facilitating the learning process. Since this person will be individually changing, all comments spoken by this person can easily be exchanged by another speaker. The approach of initiating metacogntive processes necessary for solving the given tasks can therefore easily be implemented in different settings.

References:

BACKMANN, C. W. (1974): Social Psychology. New York, N.Y.: McGraw-Hill, Inc.

BATSHAW, M.L. & PERRET, Y.M. (1992): Children with Disabilities: A Medical Primer (Third Edition). London: Brookes

BROWN, A.L. & PALINCSAR, A.S. (1990): Guided, Cooperative Learning and Individual Knowledge Acquisition. In: RESNICK, L. (ed.): Cognition and Instruction. Issues and Agendas. Hillsday, N.J.

CUMMINS, ROBERT A. & PRIOR, MARGOT P. (1992): Autism and Assisted Communication: A Response to Biklen. Harvard Educational Review, Vol. 6, No.2, 228-241

DAVIDSON, E.; DEUSER, R.J. & STERNBERG, R:J: (1994): The Role of Metacognition in Problem Solving. In: METCALF, J. & SHIMAMURA, A.P. (eds.): Metacognition. Knowing About Knowing. Cambridge: MIT Press

HASSELHORN, MARCUS & MÄHLER, CLAUDIA (1990): Lernkompetenzförderung bei “lernbehinderten“ Kindern: Grundlagen und praktische Beispiele metakognitiver Ansätze. Heilpädagogische Forschung, Band 16, Heft 1, 2-13

HECKHAUSEN, H. (1987): Causal Attribution Patterns for Achievement Outcomes: Individual Differences, Possible Types and their Origins. In Weinert, F.E. & Kluwe, R.H. (eds.): Metacognition, Motivation, and Understanding. Hillsdale: NJ: Erlbaum.

LAUTH, G. W. und SCHLOTTKE, P. F. (1997): Training mit aufmerksamkeits-gestörten Kindern. Weinheim: Psychologie Verlags Union.

NADDA (NATIONAL ATTENTION DEFICIT DISORDER ASSOCIATION): http://www.add.org/, 1999

PIEPER, M., GAPPA, H. & MERMET, S. (1998): Teleworking for Disabled People: Pitfalls and Ongoing Challenges. In: EDWARDS, A.; ARATO, A.; ZAGLER, W. (eds.): Proceedings of the XV. IFIP World Computer Congress, Computers and Assistive Technology ICCHP ´98, Vienna/Budapest, 438-443 (ISBN 3-85403-118-1)

PIEPER, M. & KOBSA, A. (1999): Talking to the Ceiling: An Interface for Bed-Ridden Manually Impaired Users. In: CHI 99, Human Factors in Computing Systems, 15-20 May 1999, Pittsburgh, PA, USA, ISBN 1-58113-158-5

SODIS (SOFTWARE- DOKUMENTATIONS- und INFORMATIONS-SYTEM): http://www.sodis.de/, 1999

SCRUGGS, T. E. & MASTROPIERI, M. A. (1993): Special Education for the Twenty-First Century: Integrating Learning Strategies and Thinking Skills. Journal of Learning Disabilities, 26, 392 -398.

TINZMANN, M.B.; JONES, B.F.; FENNIMORE, T.F.; BAKKER J.; FINE, C. & PIERCE, J. (1990):What Is the Collaborative Classroom?. Oak Brook, IL: NCREL

Non Visual Presentation of HTML documents for Disabled and Elderly Using Extended Cascading Style Sheets

Ph. Truillet and N. Vigouroux

IRIT UMR CNRS 5505
118, Route de Narbonne 31062 Toulouse Cedex 4
Tel +33 561 556 314, Fax +33 561 556 258, E-mail
`{truillet,vigourou}@irit.fr`
URL `http://www.irit.fr/ACTIVITES/EQ_IHMPT/handicap`

Abstract: The aim of this paper is to propose the expression of the CARE (Complementarity, Assignment, Redundancy, Equivalence) properties for non visual multimodal presentation of electronic documents within the extended Cascading Style Sheet (CSS) formalism.
Firstly the CARE properties and the CSS formalism are briefly described. Then, we present two synchronization and logical operators in order to convey the *Complementarity* and the *Assignment* properties of CARE. Some illustrations are given.

1. Introduction

Public World-Wide-Web users are accustomed to visual presentation of HTML (Hyper Text Markup Language) documents on a screen. This fact is changing with the development of the distance consultation of emails and HTML pages. So, alternatives to visual presentation are to be found to consult these electronic documents when they are accessed by phone or through an interactive voice server. These alternatives are crucial for the visually impaired persons and elderly persons.
For instance, screen readers based on aural and tactile presentation exist. These screen readers which simply “read” all the characters, use Text-to-Speech systems combined with “sounded” icons (Auditory Icons and/or Earcons) [Gaver 88], [Blattner 89] and/or Braille displays. These solutions lay out on a vocal or a tactile presentation of graphical objects (icons, dialog box, menus, text, etc.). Nevertheless, “presentation rules” are not used.

If many commercial accessibility tools allow blind and print-impaired communities to access electronic documents, they mainly “display” the ASCII text after the filtering of graphic objects and/or the adaptation of the document structure (loss of the spatial organization, for instance). Moreover, some works in the field of cognitive research [Truillet 99] have pointed that the document layout is a sense carrier and seems to increase the comprehension and memorization processes of texts.
Given this current and future importance, it is important to consider all the “dimensions” of an HTML document (the text itself but also all the semantic dimensions such as typographic attributes and the organization).

The goal is to provide blind readers with maximum information (both text and material layout wise) against minimum cognitive effort on their part. The design of a multimodal presentation relies on an effective cooperation between several output modalities. Here

cooperation is based on the results of Bernsen's study on output modalities [Bernsen 94] for the representation of information in the acoustic and tactile modes.
Another work used concerns the CARE properties between the interaction techniques that may be available in a multimodal user interface.

We want to study if the CARE properties can be conveyed within the CSS formalism. The CARE properties and the CSS formalism will be briefly described. Then, we present the synchronization and logical operators usable to model the *Complementarity* and the *Assignment* properties.

2. The CARE Properties WITHIN the CSS Formalism

2.1. The CARE Properties

CARE was defined as a formal framework to characterize and to assess the relations between several modalities. These relations concern: the *Complementarity, Assignment, Redundancy and Equivalence* between modalities which may be used sequentially, concurrently, independently or combined synergistically [Nigay 93]. The CARE properties [Coutaz 95] rely on the notions of modality and temporal relationships. Several works have mainly studied this expression tool for input modalities [Catanis 95], [Zanello 97].
In [Truillet 97b, 99], the CARE properties were used as a toolkit to design multimodal presentation systems of HTML documents.

2.2. The CSS formalism

The World Wide Web Consortium [W3C 98a, c] proposed the Cascading Style Sheet (CSS) formalism. Our work hypothesis is that this formalism is supposed to be flexible and adaptable for information presentation strategies. Then, the goal of this work is to demonstrate that it is possible to associate multimodal presentation to the interactive objects by using the CSS formalism.
Using Cascading Style Sheets rather than HTML tags extensions allows the same documents to be read with visual (on a screen), aural (with Text-To-Speech (TTS) systems), tactile (with Braille displays) or multimodal presentation without having to produce several forms of documents.
In this way, the HTML document representation and the document presentation are independent. This approach could provide improved accessibility of document for disabled people without compromising the visual design of the document.

The Figure 1a shows how the CSS specifies the presentation (Figure 1b) by assigning values to style properties.

```
<HTML>
<HEAD>
<STYLE TYPE="text/css">
<!--
H1
 {
  color: #008000;
  margin-left: 4%;
  font-family: Comic Sans MS, helvetica, sans-serif;
  font-size: 16pt;
 }
-->



</STYLE>
</HEAD>
<BODY BGCOLOR="FFFFFF">
<H1>Example with CSS</H1>
</BODY>
</HTML>
```

Tag

Typographic attributes

Figure 1a: CCS example.

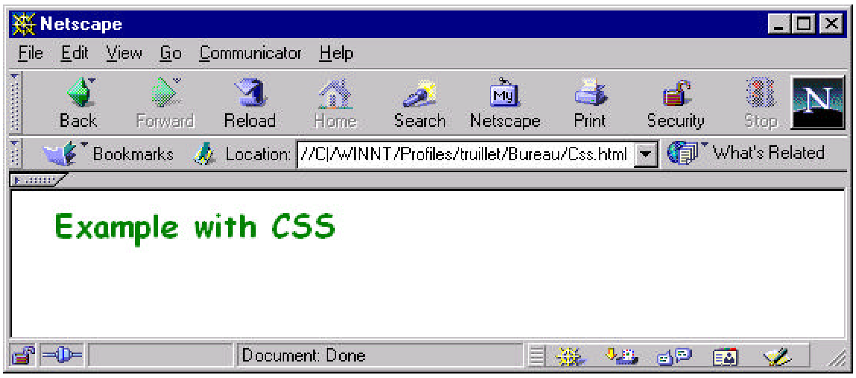


Figure 1b: Visual presentation of the description given in the Figure 1a.

Here, the visual presentation of the H1 tag is green colored, 4 inches margin-left and "Comic Sans MS" font is used.
T.V. Raman [W3C 97] proposed an extension of the CSS model for an aural presentation (***A**ural **C**ascading **S**tyle **S**heet*) by adding audio properties (*volume, pause, play*), spatial properties (*angle, elevation*) to play a 3D sound and TTS properties (*pitch, speed, voice-family*). These properties were integrated in the CSS level 2 working draft [W3C b].

2.3. CARE properties in CSS formalism

The aim of this paper is to evaluate the capabilities of the CSS formalism and its expressiveness to allow multimodal presentation. Our goal is to extend the possibilities of the CSS formalism to take into account the multimodal cooperation between tactile and aural modalities.

Assignment and *Redundancy* properties can be easily modeled in the CSS formalism as shown in Figure 2.

```
<HTML>
<HEAD>
<STYLE>
<!--
@media print, screen
 {
  H1
   {
    color: #008000;
    margin-left: 4%;
    font-family: Comic Sans MS, helvetica, sans-serif;
    font-size: 16pt;
   }
 }

@media aural
 {
  H1
   {
    cue-before: url("ding.wav");
    voice-family: robert;
    pitch: low;
    speech-rate: slow;
   }
 }
-->
</STYLE>
<LINK rel= “stylesheet” type=“text/css” media= “screen, aural”>
</HEAD>
<BODY>
. . .
</BODY>
</HTML>
```

To express the Redundancy, we list the different media.

This description conveys the Assignment property.

Assignment of output media

Figure 2: Model of the Redundancy property.

As shown in Figure 2, CSS defines presentation property classes associated on both HTML tags and media. The `@media` rule allows to extend the HTML document presentation to other media (Cf. Figure 3.).

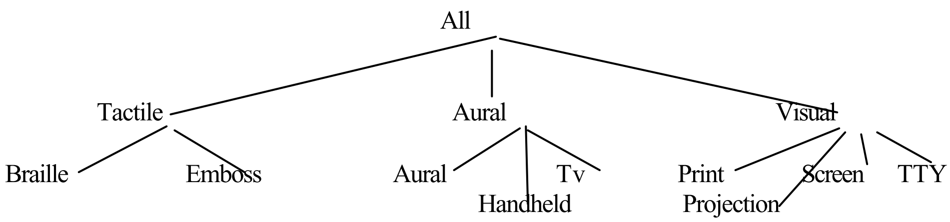


Figure 3: Media types defined within the CSS formalism [W3C 97].

For each used media, CSS specifies a possible presentation. User can modify each property on line regarding its capabilities [RNIB 98]. Nevertheless, problems still occur in order to model cooperation between modalities.

First, the *Complementarity* property is not modeled. For instance, a CSS does not allow to present typographic information on a Braille display and synthesize the information

associated on a Text-to-Speech synthesis system.
Then, the definition of the media used and its properties have to be refined and structured. For instance, the "*aural*" medium groups properties for both verbal (recorder speech, Text-to-Speech) and non-verbal (Auditory Icons, Earcons) modalities.
It will be more easy to model modalities uses than media uses.
Moreover, we have to extend the cooperation concept between modalities (in the CARE meaning) because there is no rule to model them.
Finally, temporal properties (use of several media in sequence or in parallel) are just defined in a global level. We have to refine these for each tag of the document.

3. The Extension of the CSS formalism

Firstly, we refined the different media into the modality classes as illustrated in the Figure 4.

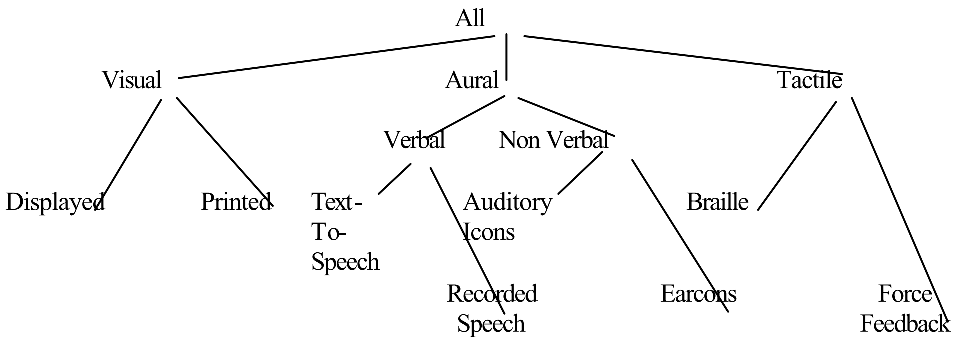


Figure 4: Modality types in the CSS formalism.

One major criterion is to stay independent of the device used. Three classes of modalities have been defined: *visual*, *aural* and *tactile*. Each class can be decomposed in several sub-classes of modalities. This proposition is not limited only to these modalities: this model can be extended to other presentation modalities.

A style is defined by its properties (for instance, size of characters for a visual presentation). This definition can be made:
- for the whole modalities,
- for each modality.

```
H1
 {
  meaning: Header 1
  // Text-to-Speech presentation
  @modality TTS:
   {
    voice-family: robert;
    pitch: low;
   }

  @modality screen:
   {
 font : Comic Sans MS;
   }
 }

  @modality braille:
   {
   }
}
```

global property (for all modalities).

specific properties (for each modality).

Figure 5: For each style, specification of the presentation.

The presentation rules are not specified necessarily even if the presentation properties have been defined. For example, how a Title (`H1`) has to be presented both to Braille display and aurally?
In consequence, we added some operators in CSS like:

1. synchronization operators between modalities; two operators were added: *parallel* noted **#** or *sequential* noted **;**. Note that synchronization and buffer-size harmonization between the Braille display and the TTS system are not easy to achieve,
2. binary operators for multimodal cooperation ("*and*" noted **&&** for synergy and "*or*" noted **||** for independence)
3. and another operator for user's preferences. This operator is defined by the order of presentation rules.

Moreover, for each used style, the presentation rules can also be refined.
In this formalism, a redundant presentation (in the CARE meaning) of a title for TTS and Braille modality is written (cf. Figure 6):

```
#(@TTS.H1 && @braille.H1)
```

Figure 6: Redundancy property for "Title" style.

User's preferences can be formalized by means of a decreasing interest list of presentation modalities as follow:

```
@braille.H1 || @TTS.H1 || @Recorded_Speech.H1
```

Figure 7: Presentation preference for "Title" style within the CSS formalism.

For example (Cf. Figure 7), the Braille modality is used firstly. By default, the TTS modality will be used and by default, recorded speech will be used.
Moreover, these preferences can be specified for each typographic attribute of the electronic document (title, bold, italic, etc.).

In this formalism, *Equivalence* is written like this:

```
@TTS.H1 || #(@braille.H1 && @Recorded_Speech.H1)
```

Figure 8: An Equivalence model.

In this example, in order to present '`H1`" style, Text-To-Speech synthesis will be used. By default, a cooperation between Braille and recorded speech will be used. This cooperation is supposed to be equivalent to TTS presentation.

The *Complementarity* property can be written as follow:

```
;(@braille.H1.style && @TTS.H1.text)
```

Figure 9: Complementary property within the CSS formalism and its extension.

The "Title" style is restituted by two modalities: Braille and TTS. Braille modality displays the style (e.g. "Title") and TTS synthesis modality the ASCII text in relation with the style.

4. Conclusion

The rewriting of CARE properties within the CSS formalism has been illustrated and has been demonstrated for multimodal presentation based on aural and tactile modalities. To permit the complementary and assignation properties, synchronization and logic operators were added. The implementation of this formalism is in progress within the Smart-Net project [Truillet 97].

An input interface is also in progress to allow inexperienced or novice users to define its presentation characteristics. After these two steps, each potential user will have the possibility to personalize the presentation style of electronic documents in regard to its behavior and perceptual model. The CARE properties and the CSS formalism are two complementary tools usable for the design of multimodal interface.

5. References

[Bernsen 94] N. O. Bernsen, A Revised Generation Of The Taxonomy Of Output Modalities, ESPRIT BRA, 7040 AMODEUS, TM/WP11, (1994).

[Blattner 95] M. Blattner, D. Sumikawa, R. Greenberg, Earcons and Icons: Their Structure and Common Design Principles, in Human-Computer Interaction, 1989, pp. 11-44.

[Catinis 95] L. Catinis, J. Caelen, Analyse du comportement multimodal de l'usager humain dans une tâche de dessin, IHM'95, Toulouse, 11-13 octobre 1995, pp. 123-129.

[Coutaz 95] J. Coutaz, L. Nigay, D. Salber, A. Blandford, J. May, R. Young, Four Easy Pieces for Assessing The Usability of Multimodal Interaction: The CARE properties, InterAct'95, Lillehammer (Norway), June 1995, pp. 115-120.

[Gaver 88] W. Gaver, *Everyday Listening and Auditory Icons*, Unpublished Doctoral dissertation, University of California, 1988.

[Nigay 93] L. Nigay, J. Coutaz, A design space for multimodal interfaces: concurrent processing and data fusion, InterCHI'93, ACM, new York, pp. 172-178.

[RNIB 98] Royal National Institute for Blind, http://www.rnib.org.uk/css.htm.

[Truillet 97] Ph. Truillet, B. Oriola, N. Vigouroux, Multimodal Presentation as a Solution to Access a Structured Document, Access Review, Spring 1997, Vol. II, N° 1, pp. 17-22.

[Truillet 97b] Ph. Truillet, Étude de l'interaction multimodale dans une tâche de lecture de documents par des déficients visuels, IHM'97, Poitiers, 10-12 septembre 1997, pp. 135-142.

[Truillet 99] Ph. Truillet, N. Vigouroux, Étude des effets de la mise en forme matérielle de textes par des présentations non visuelles, CIDE'99, Damas, 5-7 Juillet 1999.

[W3C 97] Aural Cascading Style Sheet, W3C Working Draft 30-June-1997, http://www.w3.org/TR/WD-acss

[W3C 98] World Wide Web Consortium, http://www.w3.org/

[W3C 98b] World Wide Web Consortium, "HTML 4.0 Specification", http://www.w3.org/TR/REC-html40/.

[W3C 98c] Cascading Style Sheets, level 2, W3C Working Draft 28-Jan-1998, http://www.w3.org/REC-CSS2

[W3C 98d] WAI Accessibility Guidelines: Page Authoring, W3C Working Draft, 18 September 1998, http://www.w3.org/TR/WD-WAI-PAGEAUTH/

[Zanello 97] M. L. Zanello, *Contribution à la connaissance sur l'utilisation et la gestion de l'interface multimodale*, Thèse d'Université, Grenoble, Juillet 1997.

USER INTERFACES FOR VISUALLY IMPAIRED PEOPLE

Robert Batùšek and Ivan Kopeèek
Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
e-mail: xbatusek@fi.muni.cz, kopecek@fi.muni.cz,
http://www.fi.muni.cz/~xbatusek/, http://www.fi.muni.cz/~kopecek/

Abstract

Some aspects of the user interfaces and spoken language dialogue systems developed for visually impaired users are presented in the paper. First, we discuss specific requirements of visually impaired users to user interfaces of applications and dialogue systems. The paper further deals with some specific elements of the dialogue system design that can be used to enhance the effectiveness of the communication between blind user and dialogue systems.

Next, the basic communication module used in the dialogue systems we are presently developing is outlined. We also present a brief description of two applications developed by the Natural Language Processing Group at the Faculty of Informatics, Masaryk University Brno; the dialogue programming system DIALOG and the speech oriented hypertext system AUDIS. Both systems are developed primarily for visually impaired users, especially for blind programmers and students.

1. Introduction

Although present software systems are often very sophisticated and user-friendly, they are usually not very convenient for visually impaired people. The reason is in the graphical interface and absence of the features fulfilling special needs of the blind people. Speech synthesizer and "screen access" ("screen reader") software still represents basic facilities that are used by blind users to obtain information by means of computer.

The present development in human-computer interaction and spoken language dialogue systems (especially multi-modality and progress towards less error-prone speech recognition) brings new hopes and expectations and also new problems. The design of suitable dialogue strategies which seems to be one of the crucial points of the dialogue systems development is even more important for visually impaired people, supporting perspicuity of the visually impaired user. This aspect is emphasized by the fact that for visually impaired people computers are one of the most important sources of information.

2. Specific Demands on User Interfaces for Visually impaired

In some applications, there is almost no difference in using the user interface between sighted and visually impaired user. This is, for example, the case of dialogue systems that are accessible via telephone. However, many systems use graphics as an important output information and they mostly also do not assume that the user is visually impaired and therefore they ignore specific needs of such users. Let us first summarize basic specific needs of the visually impaired user:

- The system must enable comfortable control by means of combination of speech commands and keyboard (hot-key commands). Of course, graphics can be used only in a special way as an additional output for the users that are not totally blind. Other input/output devices can be used for a specific application provided that such a type of communication is effective.
- Speech commands should be supported by speech (system driven) command dictionary that allows to express a command in several ways, making the control of the system more intuitive. This is always useful, but especially for blind users where the possibility of intuitive control of the system is even more important.
- Easy customization and configuration is very important feature of the system, especially for blind users that often use the system for a long time. This is related to the control commands, mode and type of speech synthesis output, information data structure, and other properties and options of the system.
- It is very important to enable the user to obtain the information quickly and to allow to get an informational overview. This feature is supported by various output speech modes and output speech rates as well as by speech summaries, audio glances, earcons and environmental sounds.
- The orientation of the user should be supported by the information about the position that is always accessible in speech form as well as in the form of audio glances, earcons and environmental sounds (see next section).

The last two points are closely related to the problem of the design of dialogue strategy and we will discuss them in more detail below.

3. Specific Elements of the Dialogue Design of the Blind User Oriented Systems

As we have pointed out, one of the most important problems we meet when designing spoken language dialogue systems for blind users is how to convey a variety of pieces of information (some of them can take basically graphical form) quickly and how to supply sufficient information providing full orientation of the user. The main way how to manage this is, of course, to use sound. It can be done in the form of:

- Synthesized voice produced by the syllable based speech synthesizer. This type of sound output can be used for generating output messages and reading text data. The used speech synthesizer should apply basic prosodic features to enhance quality of the speech output and to distinguish various types of utterances. It should be also able to use various voice types that can be configured by the user. Various voice types can be used to distinguish various types of information.

- Sampled voice, which can be used for all feedback messages to the user. Various types of sampled voices are used to help the user to distinguish various types of messages.
- Sound generated by the sound synthesizer, MIDI, wave tables or special samples. This type of non-speech sound is used first for environmental sounds applied to provide feedback to user actions [2], [3], secondly it can be used for earcons (non-speech glances used to give the blind user an overview by listening, [9].

Non-speech sound and flexible use of various type of the speech type can essentially help to speed up the communication. On the other hand, it can also confuse the user, if he or she is not familiar with the corresponding meaning. Hence, we propose the strategy that consist in:

- The system has to find out whether it communicates with a beginner or experienced user. It can be done either by detecting it from user reactions or by user declaration;
- Depending on the established user experience measure the system chooses the corresponding strategy, i.e. communication level that combines explicit and implicit information.
- In any time a user can use the command EXPLAIN which explains the meaning of the implicit information and therefore learn it by using the system.
- The system checks the communication; if it detects that the user shows tendency to use EXPLAIN too frequently (or if the user does not use EXPLAIN at all), it switches the communication level.
- The user can switch off the above-described "regulation" to enable learning mode or to set the communication level.

4. The Structure of a Speech Interface for Blind Users

The communication strategy described above has to be supported by the module which forms the interface between the dialogue oriented application and hardware devices and speech synthesizer and recognizer tools. Structure of such module is outlined in Figure 1. The interface has modular structure and uses some previously developed tools: the TTS system DEMOSTHENES [6], command recognition module RCG and the prosody detection module [7]. The interface is designed to be reusable in various kinds of applications, especially in the DIALOG system (see next section) and other applications for visually impaired users.

An application communicates with the main module. It sends to the interface simple requests like *Say 'Hello, world'* or *Play 'asound.wav'*. The interface informs the application about executing its requests. The application can pause (and resume) speech at any time. The interface also sends messages to the application when it recognizes voice command.

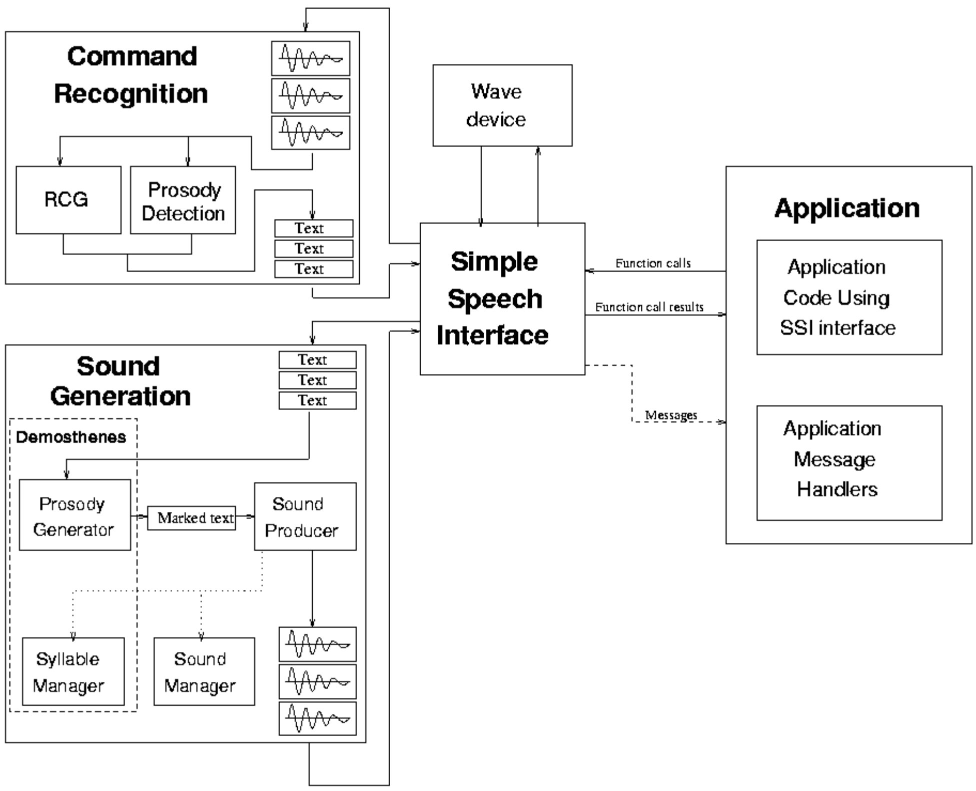
The main module of the interface operates with the sound device and guarantees that it is opened for input and/or output whenever it is needed. An application requests to perform some operations by calling a simple function. The main module only processes the request and delivers it to the appropriate submodule. The control is then returned to the application. The submodule working in a separate thread executes the request and informs the main module about the current state of the execution. The main module can inform (depending on the configuration) the application using a message. The application can use a message handler to dispatch the message.

Command recognition module continuously "listens" to the input device (usually a microphone) and tries to recognize a command. When it detects a correct command stored in its database, it informs the main module. The main module generates the appropriate message and sends it to the application.

The main module also cooperates with the prosody detection module that distinguishes three types of intonation: rising, level and falling. A command spoken with the rising intonation is understood to be a demand for assistance. A word spoken with the level intonation is interpreted as a command that will be followed by a list of parameters (or one of these parameters when such a command already has preceded). A word pronounced with the falling intonation forms a command without parameters or the last command parameter.

Sound generation module processes the texts placed to the submodule queue by the main module. These tagged texts include requests for TTS system to synthesize a text, configuration of the synthesizer (changes of the current voice, rate, volume etc.), requests for playing audio files or for mixing audio and speech. Sound manager module can either load an arbitrary audio file or one of the predefined sampled sounds (or speech messages) stored in the internal module database. It could also synthesize sounds using soundcard capabilities.

Figure 1. *Structure of the DIALOG Speech Interface System*



5. Applications

Several applications intended for blind people are currently developed, e. g. the Emacspeak system [12] or audio www browsers pwWebBrowser [10] and the IBM Home Page Reader [11]. Their main purpose is to read the textual contents of the screen more effectively than a screen-reader software is able to do. Audio web browsers usually use options provided by W3C consortium as a part of its Web Accessibility Initiative specification. Frequently used features are reading 'ALT' attribute for images, 'SUMMARY' attribute for tables or using extensions defined in CSS specification. However, such systems have some drawbacks: they do not use spoken commands, they do not involve a user model and above all they do not include any dialogue processing component. In fact, they cannot be called dialogue systems. Applications described below should overcome these insufficiencies. Actually two applications are briefly presented – the programming system DIALOG and the speech oriented hypertext system AUDIS.

The programming system DIALOG [4] is developed with the intention to provide blind user programmers (blind people do write computer programs!) with a programming language and integrated development environment adapted to their special needs. The system is designed to suppress syntactical and also some semantical errors as much as possible. In this way blind users are able to orient themselves as quickly as possible in the source code.

All the communication of the user and the system is accomplished via dialogue. A dialogue in a standard form supports source code generating, editing, testing, searching, debugging etc. The conversation is fully supported by speech input and output. Non-speech sounds are used to speed up some often used actions and subdialogues non-speech sounds are used. Source code and data have the consistent structure. Basic entities of the system are *objects*. Objects represent user-defined data or commands of the provided programming language, such as conditions, cycles, calling external functions etc. Objects are organized into a tree. To each object a set of *attributes* is assigned depending on the command which is represented by the object. DIALOG objects are designed to facilitate implementation of the dialogue processing module that prevents blind users from making errors. Each object has a name. DIALOG source code tree is naturally represented in its computer form. However, whenever the textual transcription of an object tree is needed, we use the XML data format. In what follows a classical simple program "Hello, world" written in DIALOG is presented:

```
<?xml version="1.0"?>
<!DOCTYPE PROGRAM SYSTEM "dialogprogram.dtd">
<PROGRAM Name="Hello, World">
     <FUNCTION Name="main">
          <CALL     Function="TELL"     Arguments="'Hello,
world'"/>
     </FUNCTION>
</PROGRAM>
```

For more detailed information see *http://www.fi.muni.cz/lsd/dialog/*.

The hypertext browser AUDIS ([1], [5]) is being developed since 1998. It is primarily intended for visually impaired students. One of the most serious problems for the students with sight impairment is to acquire studying materials in the appropriate form. AUDIS should allow these students to access the studying materials quickly and with respect to their needs.

AUDIS cooperates with the Interface module to perform speech communication. It uses all of its capabilities (voice commands, speech synthesis output, earcons and environmental sounds) to provide the blind users with structured textual information quickly and clearly. Special attention is paid to facilitate blind user orientation in the hypertext document structure. AUDIS also supports special graphical output for people that are not totally blind. The system creates the hypertext structure of the document that supports blind user orientation in the document. Document is divided into chapters, sections and subsections. These components are organized into a tree. Each component contains a short summary at the beginning. All chapters, sections and subsections are linked to the previous and next item at the corresponding level of the tree. Following these links user can traverse the document linearly. Moreover, information about the current chapter, section and subsection is accessible at any time. Currently, AUDIS documents have to be manually prepared.

HTML has been chosen as the AUDIS document format. The format is relatively simple, widely used and has an open standard. Thanks to this conception, extension of AUDIS functionality (which is planned in the future) to the World Wide Web browser will be quite straightforward.

Conclusions and Future Work

Developing the user interfaces and spoken dialogue systems oriented to visually impaired people brings specific problems that can be partially solved by adapting of the dialogue strategy. In the future work we would like to perform further testing of the possible extensions and modification of the method described in the paper (especially by WOZ simulation) to verify and enhance the effectiveness of the method. The development of the communication module is directed towards enabling more freedom in communication. This should be achieved by implementation of the language syntactic parser and semantic analyser [8].

Acknowledgement

The authors are grateful to K. Pala for reading the draft of the paper and valuable comments. The research has been partially supported by Czech Ministry of Education under the Grant VS97028 and by Grant Agency of the Czech Republic under the Grant 201/99/1248.

References

[1] L. Bártek, R. Batùšek, M. Fikera, I. Kopeèek: *Using Dialog Programming for Developing the Speech Oriented Hypertext AUDIS;* Proceedings of TSD'98, 1998, pp. 361-364.

[2] Darvishi, A.: *World Wide Web Access for Blind People: Problems, Available Solutions and an Approach for Using Environmental Sounds;* Proceedings of ICCHP'96, pp. 369-373.

[3] Kahlish, T.: *Improving Access to Hypertext Based Study Material for the Blind;*

Proceedings of ICCHP’96, Linz, 1996, pp. 229-236.

[4] Kopeèek, I.: *Dialog Based Programming;* Proceedings of the First Workshop on Text, Speech and Dialogue - TSD'98, 1998, pp. 407-412.

[5] Kopeèek, I.: *The Architecture of the Blind User Oriented Hypertext AUDIS;* in Proceedings of ICCHP'98, Wien-Budapest, September 1998, pp. 215-223.

[6] Kopeèek, I.: *Syllable Based Speech Synthesis;* Proceedings of the 2nd International Workshop Speech and Computer - SPECOM'97, Cluj-Napoca, 1997, pp. 161-165.

[7] Kopeèek, I: *Syllable Based Approach to Automatic Prosody Detection; Applications for Dialogue Systems;* Proceedings of the Workshop on Dialogue and Prosody Eindhoven, 1999.

[8] Pala, K.: *Semantic Annotation of (Czech) Corpus Texts;* Proceedings of the Workshop on Text, Speech and Dialogue - TSD'99, 1999.

[9] Stevens, R.D., Wright, P.C., Edwards, A.D.N., Brewster, S.A. (1996) *An Audio Glance at Syntactic Structure Based on Spoken Form;* Proceedings of ICCHP’96 1996, pp. 627-635.

[10] The Productivity Works pwWebSpeak: http://www.prodworks.com/productsindex.htm

[11] IBM Home Page Reader: http://at.rsb.org.au/products/ibmhpr.htm

[12] Raman T. V.: *Emacspeak – A Speech Interface*; Electronic Proceedings of CHI’96, http://www.acm.org/turing/sigs/sigchi/chi96/proceedings/papers/Raman/paper.html, 1996

Tailorability and Usability Engineering: A Roadmap to Convergence

Alex Totter, Chris Stary

Department of Business Information Systems
Communications Engineering
University of Linz
Freistädterstraße 315
A-4040 Linz
Austria
Email: totter@ce.uni-linz.ac.at
URL: http://www.ce.uni-linz.ac.at

ABSTRACT

Tailorability enables user interfaces to be adapted to particular needs of end users and organizations. When developers target towards the construction of user interfaces providing that kind of flexibility, they experience support at the level of testing rather than designing. In search for design support usability evaluation techniques might be used to provide early feedback from end user to designers. In this paper we put methodological and conceptual knowledge from usability evaluation into the context of designing tailorable user interfaces. In doing so, tailorability is related to existing principles of human-centered design. As a consequence, tailorability in its very technical nature has to be considered as an enabling feature for individualization. Thus, it facilitates the implementation of principles of human-centered design. Becoming part of design tools ensures final convergence of tailorability and usability engineering.

Keywords: Tailorability, usability engineering, comparative analysis, conceptual framework, definition.

INTRODUCTION

Tailorability is traditionally described as a feature of interactive software that allows the change of certain aspects of the software in order to meet different user characteristics and requirements. It is widely agreed that the design of tailorable systems is an important future challenge (Stiemerling et al., 1997; Mackay, 1991; Dangelmaier et al., 1999; Malone et al. 1995), since tailorable systems support the structured recognition of the complexity and dynamics of tasks and organizations, as well as of inter- and intra-individual differences between end users.

Tailoring activities can be performed at different system components, requiring different knowledge / understanding of the users carrying out the tailoring activity

(Mørch, 1997, Ulich et al., 1991). Sample interactive software systems that support different demands of end-users to tailor user interfaces according to their needs are: OVAL (Malone et al., 1997), the HyperCard programming system for the Macintosh, and spreadsheet applications in general.
However, several authors have pointed out that the design of tailorable interactive systems lack development procedures, frameworks, commonly agreed concepts, and effective support, since designers have to take into account several problems which classical design methodologies do not address (Kahler, 1995; Appelt et al., 1997, Stiemerling et al., 1997; Mørch et al., 1997). In this paper, to overcome these deficiencies, we put methodological and conceptual knowledge from usability evaluation into the context of designing tailorable user interfaces. The roadmap to convergence is enabled through evaluation activities becoming part of design activities. Evaluation activities thus should provide early feedback from end users to designers and foster direct input to developers. Our presented roadmap to convergence does not only contain an analysis of terms and interpretations of tailorability, but also a framework how to analyze existing usability evaluation instruments with respect to structured design support and design tool development.

In this paper we particularly
(i) analyze existing interpretations of tailorability
(ii) derive characteristic properties from the results of (i) in a structured way
(iii) contrast and supplement the results of (ii) with the results of a structured analysis of evaluation techniques addressing usability design principles.

The structure of the paper is as follows: We first review related work in the subsequent section with respect to the steps listed above, linking design activities for tailorable systems with evaluation activities. Steps (i) and (ii) are captured in the section on Capturing the Polymorphism of Tailorability. In the section on The Roadmap To Convergence With Usability Engineering we start out with analyzing existing usability engineering techniques and principles, and their contributions to implement or evaluate tailorability, thus, preparing to implement step (iii). We subsequently compare these results with the elements found in step (ii) and come up with an integration of results. Finally, we discuss the results from (ii) and (iii) with respect to the development of design support tools to construct tailorable software systems.

RELATED WORK

In the following we first review work that has already been performed on analyzing tailorability. This part addresses steps (i) and (ii) of our procedure given in the introduction of the paper. We proceed with related work in the field of analyzing usability evaluation techniques. This part addresses the initial task to be performed in step (iii) given above. Finally, we give evidence that evaluation and design activities have to be intertwined, in order to build tailorable systems.

Towards a Framework for Content Analysis. In search for a structured (re)presentation of understandings and interpretations of tailorability, the most comprehensive study has been performed by Ulich et al. (1991). They classified tailoring activities along two dimensions, identifying (common) characteristic properties of tailorable systems as follows:

Modifications of system components:

- input devices: input devices can be selectable (e.g. mouse or keyboard);
- screen layout: size, positions, colors, etc. of windows, menus, texts etc. can be modified;
- data display mode: display information can be changed qualitatively; information can be shown or hidden, alternative display modes (text/graphic) are selectable;
- scope of commands: commands can be included or excluded from a predefined repertoire, but the definition of new commands is not possible;
- command structure: new commands can be developed (e.g. by means of a macro facility);
- individual applications: new applications with dialog structures can be developed;

Methods for accomplishing user tailorability:

- selectable alternatives: different procedures for accomplishing a goal are implemented into the system in a parallel fashion and can be selected (but not modified) by the user;
- online configuration commands: user tailorability can be accomplished by means of special commands included in the application;
- configuration program: modifications can be accomplished interactively by means of a special program or module outside the application;
- configuration file: modifications can be accomplished by editing a special configuration text file with a text editor.

Although tailorability is mainly defined as a technical feature, the structure tells us something about the WHAT has to be achieved with respect to tailorability – see *modifications* part above, and the HOW the addressed properties can be implemented – see *method* part above. Hence, any comparative analysis of tailorability should distinguish between semantic and pragmatic facets.

A Framework for the Comparative Analysis of Usability Evaluation Techniques. There exist few frameworks to compare usability engineering techniques with respect to several perspectives. Most of the existing schemes try to classify techniques for evaluation in very general terms, such as Wixon et al. (1997), McGrath (1995), traditionally based on empirical terms, or to focus on particular, more or less arbitrarily selected, aspects, lacking structural considerations, such as Lewis et al. (1997), and Oppermann et al. (1997). However, a particular multi-perspective framework has been designed to demonstrate the coverage of the usability evaluation space, and to indicate differences on how to evaluate usability: Wixon et al. (1997, p. 681ff.) introduce several dimensions along which usability evaluation techniques can be classified. The authors point out that usability is a multifaceted concept that depends on users, the environment and task characteristics. These parameters allow to categorize evaluation techniques according to the following dimensions and ranges of values:

- ***Formative versus summative methods***: Formative methods are used to create a design, whereas summative methods are used to evaluate a design.
- ***Discovery versus decision methods***: Discovery methods are aiming at discovering how users work, behave, or think and what problems they have. Decision methods are used to structure interface designs or to choose between dialog elements. This polarization is sometimes called qualitative (i.e. discovery) in contrast to quantitative evaluation (i.e. decision).
- ***Formalized versus informal methods***: Some methods are highly structured and have been described formally.
- ***User involvement versus user exclusion***: Methods differ to the extent to which users are involved in evaluation, analysis and design.
- ***Component evaluation versus complete evaluation***: Some methods cover all the steps required to complete designs with respect to usability.

Unfortunately, this categorization neither provides much information in how far the evaluation techniques provides adequate feedback for the design of tailorable systems, nor are the addressed dimensions logically independent, as it could be exemplified for the techniques analyzed in the course of this study (see also Table 1). In practice, techniques tend to overlap in certain areas defined by these dimensions. Hence, for the objectives of our study, a novel framework for analysis and structured representation had to be developed.

Bridging the Gap between Evaluation and Design. Few approaches to support the ***design*** of tailorable systems have been introduced, such as participatory and evolutionary design involving the users of an application actively in the design process and giving them the opportunity to articulate their requirements. As part of these techniques early usability testing is introduced to evaluate the usability of a software system (Hackos & Redish, 1998). However, the existing literature does not report on usability evaluation techniques that actively support the design of tailorable systems, e.g. through automatically executing guidelines stemming from a styleguide, as soon as specification starts.

The particular recognition of the lack of (pro)active design support motivated us to overcome deficiencies to implement the ideas of tailorability early in the development process. In order to develop techniques and tools supporting the design of tailorable systems the developed roadmap is essentially based on

(i) a classification and categorization of existing approaches to achieve tailorable systems – the list of properties is given at the end of the following section, and
(ii) a structured representation of existing usability evaluation techniques (referring to tailorability issues).

This procedure allows us to conclude with a set of requirements for proper tool development.

CAPTURING THE POLYMORPHISM OF TAILORABILITY

Scanning the literature, several descriptions of tailorability can be found: For instance, Appelt et al. (1997) state, that "... the interface of a computer system is called tailorable when users can ***adapt*** the interface (and the system behind the interface) to their particular situation and their particular needs". To perform this activity, the tailoring of a user interface is described as some sort of a programming task which requires certain programming skills of the tailoring user, classifying users, according to their programming skills into workers, tinkerers, and programmers (introduced by MacLean et al. 1990).

Mackey (1991) refers to tailorability in terms of ***customization*** software defined as "... having mechanisms that allow users to customize their personal software environment without writing code, with changes that persists across sessions". This kind of tailoring has been identified as being most frequently performed, for example, setting default preferences for user interface layout and configuration options.

Other authors introduce the term ***end user programming***, to describe tailorability as "... system development undertaken by users to further develop an existing system to needs that were not accounted for in the original system" (Mørch, 1997), extended to reach software below the user interface and allow different levels of a system to be tailored through extensions, by writing new program code.

Malone et al. (1995) let end users become designers by giving them end-user programming tools, calling them "radically tailorable systems, allowing end users to create a wide range of different applications by progressively modifying a working system. They "... use the term 'tailorable' to mean that these systems can be changed without ever 'really programming'. More specifically, by tailorable we mean that end users can progressively modify a working system without ever having to leave the application domain to work in separate underlying 'programming' domain".

Only a few authors relate the technically oriented concept of tailorable user interfaces to software ergonomic principles, such as flexibility, to support user-oriented system design. Kahler (1995) and Ulich et al. (1991) suggested to introduce the tailorability approach ".. to reach the necessary flexibility of information technology" in general.

Briefly summarized, the understanding of tailorability can be characterized in general as to address the modifications/changes of the functionality, look and feel of an interactive software system, capturing different user requirements as well as complex and dynamic work settings (such as changes in task accomplishment and organizational settings). Synonyms have been identified, referring to the concept of user interface tailorability as customization or adaptation.

Moreover, different levels of adjustment for different needs and qualifications are suggested (Kahler, 1995), such as recording of macros in word processors to automate sequentially executed tasks; the implementation of an access policy using mechanisms for discretionary access control or changing the screen to the current user’s favorite color.

Taking into account the findings by Ulich et al. (1991), namely considering the structure telling us something about WHAT has to be achieved with respect to tailorability – (modifications part of the study), and the methods telling us something about HOW the addressed properties can be implemented (method part of the study), we are able to come up with a descriptive framework of tailorability. It comprises the following elements: synonyms, objectives, roles, features, and levels of adjustment. In the following we exemplify assignments of findings to each of the elements of the framework:

Synonyms:

- adaptation or
- customization.

Objectives:

- meeting user needs;
- persistence of changes across sessions;
- progressively modifying working systems.

Roles involved:

- primarily end users qualifying them as developers, and
- developers enabling the qualification and instantiation process.

***Features* of the interactive** software system with respect to tailorability

- non-programming means (no code writing) ;
- progressive mechanisms.

Levels of adjustment:

- system components: input/output devices, scope and structure of commands etc.;
- interaction modalities.

THE ROADMAP TO CONVERGENCE WITH USABILITY ENGINEERING

In order to compare the results from the descriptive analysis of tailorability with techniques from usability engineering, we have to analyze existing usability engineering techniques. We will start with in how far their underlying usability evaluation principles refer to tailorability. We proceed with the structured representation and analysis of some of the major usability engineering techniques, providing the roadmap of convergence.

Usability Engineering – Fundamentals

Usability has been referred to as the quality of a product in use and is defined in the ISO 9241-11 standard (ISO, 1997) as follows:

- Usability of a product is the extent to which the product can be used by ***specified users*** to achieve ***specified goals*** with effectiveness, efficiency, and satisfaction in a ***specified context of use***.

Karat (1997) elaborates upon the definition given above:

- The usability of a product is not an attribute of the product alone. It is rather an attribute of interaction with a product in a context of use.

- A usability evaluation technique is a process for producing a measurement of usability.

These explanations identify usability as a complex, multidimensional concept, requiring the integrative consideration of cognitive components (specified users), the organization of the environment (specified as a set of goals in a specified context of use, such as work), technical features (the product), and their intertwining (interaction). In order to capture these requirements, more concrete usability evaluation principles have been developed as a basis for implementing the measurement of usability. They have been incorporated in usability engineering techniques. Hence, we will analyze usability evaluation techniques focusing on two questions:

(i) Which of their underlying usability principles can be correlated to the derived characteristics of user tailorable systems (see previous section) to provide adequate input for the design of tailorable systems?
(ii) In how far is the structure of the evaluation technique (roles, activities etc.) appropriate to generate inputs for system design?

Usability Engineering – Techniques for Evaluation

The selection of the techniques has been performed according to their objective, namely measuring usability, and their common availability. After a brief description of each of the selected techniques, a widely used classification framework is applied to categorize the techniques' coverage of the usability evaluation space, thus justifying their selection. However, as already discussed in the section Related Work, this framework does not meet the objectives of this study. Hence, we will continue with the descriptive framework introduced in the previous section to complete the study.

The selected techniques for usability evaluation are:

- ***Cognitive Walkthrough*** (Lewis & Wharton, 1997): This usability inspection method focuses on ***evaluating a design for ease of learning***. It attempts to provide a detailed, step by step evaluation of the user's interaction with an interface in the process of carrying out a specific task. The process of the cognitive walkthrough includes a preparation phase, the analysis and the follow up phase. In particular, the analysts ask the following four questions:
    - Will the user try to achieve the right effect?
    - Will the user notice that the correct action is available?
    - Will the user associate the correct action with the effect that user is trying to achieve?
    - If the correct action is performed, will the user see that progress is being made towards solution of the task?

  The technique seeks to identify mismatches between users' and designers' conceptualization of a task, poor choices of wording for menu titles and button labels, and inadequate feedback about the consequences of an action. The method is performed by analysts and reflects the analysts' judgments. They identify problems by tracing the likely mental processes of a hypothetical user.

- ***EU-Con*** (Stary et al., 1998): The EU-CON technique has been developed to implement the ***EU-directive 90/270/EEC*** (EU, 1990) on man-machine communication. According to the directive the software has to meet several minimal requirements, namely:
  - task conformance,
  - ease of use,
  - ***adaptability towards human capabilities, skills and experiences,*** and
  - support of human-information processing.

  Adaptability is understood as the provision of mechanism to enable developers and end users to react on dynamically occurring requirements. These requirements might concern tasks, user characteristics, interaction modalities or a combination of those items. In addition, an interactive software system is considered to be adaptable to individual needs (based on the principle of individualization) in case it can reflect user needs and skills with respect to a certain task.

  EU-Con follows a two step strategy. First, the users are guided through a measurement procedure, in order to indicate usability problems. Then, evaluators identify the reasons for the problems users are experiencing, and try to remove barriers to effective and efficient interaction. The procedure consists of four phases: preparations, execution, tuning, and rework. Within each phase several steps have to be performed. The evaluation process is supported be a questionnaire to be filled in by users, a guide for evaluation and a handbook for evaluation and engineering.

- ***ISO 9241 evaluator*** (Oppermann et al., 1997): This technique is an example of a expert-based evaluation method for ***conformance testing with the ISO 9241 standard part 10-17***. The criteria of the ISO 9241 standard part 10-17 are:
  - suitability for the task;
  - self-descriptiveness;
  - controllability;
  - conformity with user expectations;
  - error tolerance;
  - ***suitability for individualization***: "Dialogue systems are said to support suitability for individualization if the system is constructed to allow for adaptation to the users' needs and skills for a given task....The overall objective should be to provide mechanisms which allow the system to be adapted to the individual
    - knowledge and experience of the computer
    - knowledge and experience of task domain
    - language and culture
    - perceptual / motor abilities
    - cognitive abilities of the user"
  - suitability for learning.

The ISO 9241 evaluator is a guideline-oriented evaluation technique that tests the multiparty standard ISO 9241 in about 300 items. The test items are structured in a two dimensional space defined by technical components and software ergonomic principles. Each item checks a particular aspect of ergonomic requirements specific for the given component and principles. The primary scope of the evaluation with the ISO 9241 evaluator is the user interface of a software system. The ISO 9241 evaluator offers support for testing, for documenting the testing, the evaluation, and for reporting the results. To collect information about the context of use, a simplified workplace analysis and a questionnaire exploring user characteristics have to be administered.

- ***Heuristic Evaluation*** (Nielsen, 1994): This technique is a usability engineering method for locating usability problems in user interface design in a way that it can be performed as part of an iterative design process. Heuristic evaluation involves the participation of a small group of usability experts, who ***examine the interface and judge its compliance with recognized usability principles***, so called usability heuristics:
  - visibility of the system status;
  - match between system design and the real world;
  - user control and freedom;
  - consistency and standards;
  - error prevention;
  - recognition rather than recall;
  - ***flexibility and efficiency of use:*** Accelerators - unseen by novice users - may often speed up the interaction for the expert user to such an extent that the system can cater both inexperienced and experienced users; Allow users to tailor frequent actions.
  - aesthetic and minimalist design;
  - help users recognize, diagnose, and recover from errors;
  - help and documentation.

  In the course of an evaluation session the evaluator studies the user interface several times, inspects the various dialog elements, and checks their structure and behavior against a list of recognized usability principles. Evaluators are supplied with a typical scenario of use, listing the various steps a user would take to perform a sample set of realistic tasks. Such a scenario should be constructed on the basis of a task analysis performed with actual users under actual work conditions, in order to generate utmost representative scenarios. The result is a list of usability problems in the interface, with reference to those usability principles that were violated by the design.

Having briefly outlined the techniques under investigation, we will now cluster them in Table 1 according to Wixon et al. (1997, p. 681ff.) to justify their selection. As it can be seen the selected techniques do not only address tailorability (level of principles), but also capture a large variety of known methods for evaluation (level of method).

Table 1: Categorizing the Techniques under Investigation

Method Classification	Cognitive walkthrough	EU-Con	ISO 9241 evaluator	Heuristic Evaluation
Formative Summative	Formative	Formative Summative	Summative	Summative
Discovery Decision	Discovery	Discovery Decision	Decision	Discovery Decision
Formalized Informal	Formalized	Formalized	Formalized	Informal
User involvement Expert-based	Experts	User involvement	Experts	Experts
Complete Component	Component	Component	Component	Complete

The Roadmap – Structured Representation

To enable the analysis of usability evaluation techniques according to the objectives of the study a structured scheme of representation has been developed. The scheme comprises (i) the activities to be performed in a certain sequence, (ii) the tools that have been developed to support the particular evaluation technique, and (iii) the persons (roles) that perform the evaluation. Based on this information, a comparison with the results from the content analysis can be performed.

Cognitive Walkthrough: The purpose of this technique is to suggest to the designer where the design is likely to fail and why. It examines a design for ease of learning. The analysts think about the mental processes of users. Since end users are not involved, the validity of the analysts' assumptions about the mental models of the end users are not compared to the end users' mental models.

The focus on the tailorability is not given, since the designer is responsible for determining ***one*** correct sequence of actions, no matter whether there actually exists more than one (best) way of performing the task. Furthermore, in addition to the four questions, the technique does not offer tools that support the analysts in how to perform the evaluation.

EU-Con: Since EU-Con has been developed according to the directive, the usability evaluation criteria ***"adaptability towards human capabilities, skills and experiences"*** has been turned into operational definitions for measurement, meeting the characteristics of tailorable systems. Through the participation of end users, and additional questions collecting information about end users, the identification of inter-individual differences is supported.

Moreover, direct feedback for the designer of a tailorable system is provided through the discussion between end users and designers on how to improve the interface, as well as the support through the handbook for engineering (which provides suggestions for improving the user interface to compensate for specific categories of problems).

Table 2: Structured representation of the Cognitive Walkthrough

Roles	Activities	Tools to support the technique
	Define inputs to the walkthrough	
Analysts	• define assumed user background	• ?
Analysts	• choose sample tasks	• task should be important, task should be realistic
Designer	• specify correct action sequences tasks	• rough guidelines, most common way
Designer	• determine interface states along the sequences	• stepping through the interface
?	**Convene the analysis**	
	Walk through the action sequences for each task	
Analysts	• for each correct action construct a success story that explains why users would choose that action	• ask four questions
	• use a failure story to indicate why a user would not choose that action	• ask four questions
	Record critical information	
Analysts	• make assumption about the user population • note side issues and design changes • tell credible success stories • tell failure stories (record problems, reasons) • consider and record design alternatives	
Designer	**Modify the interface design to eliminate problems**	

Table 3: Structured representation of the EU-Con

Roles	Activities	Tools to support the technique
	Preparation phase	
supervising evaluator	• brief the user	• briefing sheet
	• handle out questionnaires for evaluation	• questionnaire
end user	• identify task	• manual that guides the user through the entire identification and questioning-answer process
end user	**Execution phase** • complete the questionnaire	• questionnaire
	Tuning phase	
supervising evaluator	• utilize the questionnaires • interpret the results	• alert sheets • handbook for evaluation • handbook for engineering
	Rework phase	

supervising evaluator supervising evaluator end user designer	• identify possible reasons for usability problems • discussion problems in detail with end users • implement improvements at the user interface	• handbook for evaluation • handbook for engineering

ISO 9241 evaluator: This expert- and guideline-oriented technique evaluates the user interface through conformance testing with the ISO 9241 standard part 10-17. Since one of the principles focuses on the ***suitability for individualization,*** tailoring aspects are considered. In particular, through the two dimensional space, defined by technical components and software ergonomic principles, a direct relation to the modifications of system components to be tailored can be given. Unfortunately, no end-users are involved. Therefore, the suggestions made by experts concerning tailorability functions can only be given at a very general level.

Table 4: Structured representation of the ISO 9241 evaluator

Roles	Activities	Tools supporting the technique
	Preparation phase	
evaluator (expert with good expertise about human factors)	• gather typical user characteristics • gather typical task characteristics	• questionnaire • simplified work place analysis
?	• chose actual software / user interface	?
evaluator	• specify test situations for each item	• view editor
evaluator	• record test situations for each item	• ISO 9241 evaluator
evaluator	• evaluate each test item for the test situations	• ISO 9241 evaluator • EVADIS evaluation guide
evaluator	• write an explanation for the evaluation	• integrated capture tool
evaluator	• capture detected deficiencies	• integrated capture tool
evaluator	• write evaluation report	• text editor

Heuristic Evaluation: This task- and user-independent technique directs the attention mainly to the characteristics of the interface. Since one of the heuristics focus on ***"flexibility and efficiency of use"*** tailoring aspects are considered. Unfortunately, through the lacking participation of end users, only general assumptions can be made. Although in the debriefing session the outcome of the evaluation is discussed and suggestions for improving the interface can be gathered, no tools have been developed to support the aforementioned processes or any re-design (see table 5).

Table 5: Structured representation of the heuristic evaluation

Roles	Activities	Tools to support the technique
	Pre-Evaluation training session	
Heuristic evaluation expert designer?	• briefing on the method • briefing on the domain • briefing on the scenario	? ? • task analysis
	Actual evaluation	

evaluator (usability and or interface specialists) evaluator	• find as many usability problems in the interface as possible • state what established usability principle was violated by each usability problem	• scenarios • usability heuristics
evaluator	**Debriefing session to discuss outcome of the evaluation** • modify the heuristic evaluation method to include advice for the redesign phase	?
evaluator	**Severity rating phase** • assess the severity of the usability problems that had been found in the evaluation session.	• rating scale

In a brief summary we can conclude that many techniques

(i) can be related to issues of tailorability, either through their direct relationship to standards or generally acknowledged principles, or through their freedom to select particular principles, as experts feel appropriate, for the purpose of evaluation;
(ii) are incomplete, either with respect to role definitions, activity specifications, and/or tool support;
(iii) do not address the incorporation of the results of evaluation into other phases or steps of development than usability testing, although some of the authors claim to address design issues;
(iv) tend to abstract from tasks and needs from end users. They rather assume that deficiencies at the user interface can be detected by independent experts and/or task- and user-independent settings for evaluation.

In general, it has to be recognized that usability engineering is understood mainly in terms of usability testing. This fact might explain the lack of explicit and well grounded relationships to design activities.

Striving for Convergence

Based on the results from the descriptive content analysis as well as those from structuring existing evaluation techniques the following commonalties and required extensions of usability techniques become evident:

Roles involved:

- tailorability activities should be performed by end users requiring different levels of programming skills;
- some of the existing evaluation techniques do not involve end users;

From these results we can conclude:

- End users as well as developers might be part of the usability engineering process.
- However, developers have to change their roles toward enablers of qualification and functions, based on initial, direct user inputs.

Objectives

- tailorability activities should capture different user requirements as well as complex and dynamic work settings;
- usability evaluation techniques mainly measure usability based on usability evaluation principles, such as ease of learning, ISO standard 9241 part 10 ...;

From these results we can conclude:

- Some of the usability evaluation techniques measure tailorability only implicitly through very general usability principles, not focusing on different user characteristics of taking into account dynamic work settings.

Features

- usability evaluation techniques hardly measure interactive systems at the features level.

From these results we can conclude:

- Features of the interactive software system with respect to tailorability, such as non-programming means (no code writing) or progressive mechanisms can become part of evaluation, in case concrete activities, as exemplified in Ulich et al. (1991) through commands and files, become available.

Levels of adjustment

- usability evaluation techniques evaluate interaction devices and modalities at a very general level, frameworks for systematic investigations of levels of adjustments are missing.

From these results we can conclude:

- Levels of adjustment, such as proposed by Ulich et al. (1991), are currently restricted interaction devices and modalities. They have to be enhanced with the conceptually addressed ones, namely scope and structure issues.

For the development of proper design tools the following list of requirements can thus be derived:

(i) Several roles have to be provided. They range from end users to evaluators – see the structured representation of evaluation techniques.
(ii) The principle of tailorability with respect to task structure and accomplishment, user capabilities, needs, and preferences, and with respect to interaction modalities have to become part of the tool, e.g. through a hyper-linked knowledge base.
(iii) The target user population of such tools is not only the development team, but rather the end user who should become qualified to use tailorability functions.
(iv) The tool has to provide process support to achieve tailorability.

CONCLUSIONS

Becoming aware of the demand for tailorable software systems, we set out exploring the understanding of tailorability and corresponding support for design and evaluation. Our first impression has turned out to be the regular situation in software development: Although standardization bodies and usability evaluators care about features of tailorability, there is a lack of operational support for designers, who have to be considered to be the ultimate pre-processors to programmers. In case programming is not preceded by clear tailorability objectives and accurate design specifications developers have to wait until usability testing for user feedback to measure the success of their efforts, rather than being able to check their code against tailorability specifications and corresponding operational requirements.

Since developers find themselves lost with respect to tailorability design support we have developed a framework and roadmap to investigate how to take into account existing interpretations of tailorability and respective techniques, in order to develop specifications for tailorability design support tools. We have also been driving some of the roads on the map, in order to proof our methodological concept. We have also paved some of the roads, giving some results.

Although we were able to find some explicit correspondences between the two fields involved, some links might have remained implicit ones. They have to be elaborated in further studies of this kind. In particular, a more extensive literature survey has to be performed, and a larger amount of usability engineering approaches has to be represented and compared in a structured way. Moreover, concepts related to tailorability via the organization of work, the social setting of end users, and cognitive capabilities, have to be analyzed and put into the context at hand.

However, some results of the process we started can be given by that time:

- Tailorability is not merely a set of technical features to be adjusted. It requires a process, different levels of adjustment and has to take into account the entire 'reality' of end users, as perceived by them.
- Tailorable systems require design techniques and tools, with strong emphasis on contextual knowledge and the evaluation of design results.
- The design of tailorable systems requires a closer look at and involvement of the actual end users. Hence, the development process **as well as** the result require end user involvement and participation in a fully competent and qualified way.

As a consequence, the development process might not occur in a structured and straightforward way as developers might wish to experience when designing tailorable systems. A shift of roles is required. Developers have to become enablers to provide end users with capabilities for tailoring **and** qualify them using these functions, since tailorability has to be understood as end user empowerment from the very beginning of development instead of adding 'just-another' set of technical functions as part of the user interface or the application's functionality.

References

Appelt, W.; Hinrichs, E. and Woetzel. G. (1997): Effectiveness and Efficiency: the Need for tailorable User Interfaces on the Web. In: Proceedings of the Seventh International World Wide Web Conference, Brisbane, Australia, 14th-18th August, 1997.

Dangelmaier, W.; Hamoudia, H.; Klahold, R. F. (1999): Domain Preferences for End-User Tailoring in Shared Workflow Interfaces. Workshop Implementing Tailorability in Groupware, in conjunction with the International Joint Conference on Work Activities Coordination and Collaboration WACC'99, San Francisco, CA, USA, February 1999.

EU (1990): Directive 90/270/EWG on Man-Machine Communication, in: Directive 29.5.1990 about Safety and Health of VDU-Work, Official EC-Newsletter, 33, L156, Minimal Standards, Par. 3, p. 18, 21.6.

Kahler, H. (1995): From Taylorism to Tailorability. Supporting Organizations with tailorable Software and Object orientation. In: Anzai, Y.; Ogawa, K.; Mori, H.: Symbiosis of Human and Artifact. Human and Social Aspects of Human-Computer Interaction. Proceedings of the Sixth International Conference on Human-Computer Interaction, (HCI International'95), Tokyo, Japan, 9-14 July 1995, Volume 2, pp. 995-1000, Elsevier: Amsterdam.

Karat, J. (1997): User-Centered Software Evaluation Methodologies, in: Helander M.; Landauer, T.K.; Prabhu, P. (eds.): Handbook of Human-Computer Interaction (2^{nd} ed.); Elsevier Science B.V.; pp. 689-704.

Lewis, C.; Wharton, C. (19997): Cognitive Walkthroughs, in: Helander M.; Landauer, T.K.; Prabhu, P. (eds.): Handbook of Human-Computer Interaction (2^{nd} ed.); Elsevier Science B.V.; pp. 717-732.

McGrath, J.E. (1995): Methodology Matters: Doing Research in the Behavioral and Social Science, in: Baecker, R.M.; Grudin, J.; Buxton, W.; Greenberg, S. (eds.): Readings in Human-Computer Interaction: Towards the Year 2000 (2^{nd} ed.), Morgan Kaufmann Pub.: San Francisco, pp. 152-169.

McLean, A.; Carter, K.; Lövstrand, L., Moran, T. (1990): User-tailorable Systems: Pressing the Issue with the Button, in: Proceedings of the Conference of Computer-Human Interaction, CHI'90, April, 1-5, 1990. Seattle, Washington, ACM Press, New York, 1990, pp. 175-182.

Mackay, W.E. (1991): Triggers and Barriers to Customizing Software. In: Proceedings of ACM CHI '91 Human Factors in Computing Systems. New Orleans, Louisiana: ACM/SIGCHI, pp. 153-160.

Malone, T.W.; Lai, K.; Fry, C. (1995): Experiments with Oval: A Radically Tailorable Tool for Cooperative Work. In: ACM Transactions on Information Systems, Vol. 13, No 2, April 1995, pp. 177-205.

Mørch, A. (1997): Aspects of End User Tailoring: In: Workshop on Tailorable Groupware: Issues, Methods, and Architectures. GROUP'97, International Conference on Supporting Group Work: The Integration Challenge November 16-19, 1997, Embassy Suites Hotel, Phoenix, Arizona, USA.

Nielsen, J. (1994): Heuristic Evaluation, in: Nielsen, J.; Mack, R.L. (eds.): Usability Inspection Methods, John Wiley & Sons, Inc.: New York, pp. 25-62.

Opperman, R.; Reiterer, H. (1997): Software Evaluation using the 9241 Evaluator, in: Behaviour & Information Technology, 1997, Vol. 14, No. 4/5, pp. 232-245.

Stary, Ch.; Riesenecker-Caba, T.; Flecker, J. (1998): Implementing the Directive for VDU work - the EU-state of the art, in: Behaviour & Information Technology, 1998, Vol. 17, No. 2, pp. 65-81.

Stiemerling, O.; Kahler, H.; Wulf, V. (1997): How to make Software Softer - Designing Tailorable Applications. In: Proceedings of DIS'97: Designing Interactive Systems: Processes, Practices, Methods, & Techniques, 18-20 August 1997 Grand Hotel Krasnapolsky Amsterdam, The Netherlands 1997, pp. 365-376.

Ulich, E.; Rauterberg, M.; Moll, T.; Greutmann, T.; Strohm, O. (1991): Task Orientation and User-Oriented Dialog Design. In: International Journal of Human-Computer Interaction 3(2), pp. 117-144.

Wixon, D.; Wilson, Ch. (1997): The Usability Engineering Framework for Product Design and Evaluation, in: Helander M.; Landauer, T.K.; Prabhu, P. (eds.): Handbook of Human-Computer Interaction (2nd ed.); Elsevier Science B.V.; pp. 653-688.

Evaluating adaptable and adaptive user interfaces: lessons learned from the development of the AVANTI Web browser

Constantine Stephanidis, Alexandros Paramythis, Michael Sfyrakis

Institute of Computer Science
Foundation for Research and Technology - Hellas
Science and Technology Park of Crete
Heraklion, Crete, GR-71110, Greece
{cs, alpar, sfyrakis}@ics.forth.gr

Abstract. One of the critical issues in the development of adaptable and adaptive user interfaces concerns the lack of appropriate evaluation methods and techniques. Existing approaches cannot be used to assess the way and extent to which the adaptation facilities of the interface affect interaction qualities such as accessibility and usability. This paper reports on the particular approach taken within the development of the AVANTI Web browser for the assessment of the adaptation characteristics of the user interface. Based on the practical experience gained, a number of requirements have emerged towards the development of generic methods and techniques for evaluating adaptation-capable user interfaces.

1. Introduction and Background

The AVANTI project[11], which was concluded in August 1998, aimed to address the interaction requirements of individuals with diverse abilities, skills, requirements and preferences (including disabled and elderly people), using Web-based multimedia applications and services. Detailed descriptions of the AVANTI system and its main components can be found in [Fink, Kobsa & Nill, 1997; Stephanidis et. al., 1998; Fink, Kobsa & Nill, 1999].

The AVANTI Web browser acts as the front end to the AVANTI system, and aims to satisfy diverse end user abilities, requirements and preferences, as well as different usage contexts. The design and development of the AVANTI Web browser followed the Unified User Interface Development approach [Stephanidis, Savidis & Akoumianakis, 1997], which led to the construction of a unified browser interface, capable of adapting itself to suit the requirements of different user categories: able-bodied, people with light or severe motor disabilities, and blind people [Stephanidis et al., 1998].

[11] Part of the R&D work reported in this paper has been carried out in the context of the ACTS AC042 AVANTI project "Adaptive and Adaptable Interactions to Multimedia Telecommunications Applications" (September 1995 - August 1998), partially funded by the European Commission (DG XIII). The AVANTI consortium comprised: ALCATEL Siette (Italy) - Prime contractor; CNR-IROE (Italy); ICS-FORTH (Greece); GMD (Germany); University of Sienna (Italy); MA Systems (UK); MATHEMA (Italy); VTT (Finland); ECG (Italy); University of Linz (Austria); TELECOM ITALIA (Italy); EUROGICIEL (France); TECO (Italy); Studio ADR (Italy).

Unified User Interfaces employ adaptation techniques to automatically tailor themselves to different sets of user and usage context characteristics [Stephanidis, Savidis & Akoumianakis, 1997]. Specifically, the design phase involves the construction of a *polymorphic task hierarchy*, within which different tasks may have alternative instantiations in the user interface (called *instantiation styles*, or simply *styles*). The task decomposition thus proceeds in a polymorphic fashion, defining alternative styles and task hierarchies, according to requirements and preferences of different user categories. In other words, different styles define alternative ways in which a specific task can be realised.

Following Unified User Interface Design, the resulting single design artefact may have multiple instantiations during initiation of interaction (adaptability), in order to ensure accessibility for a wide range of users. Moreover, each interface instance is continuously enhanced at run-time (adaptivity), in order to provide high-quality of interaction to all potential users. In AVANTI, both adaptability and adaptivity were supported through a rule-based framework, which necessitated the transformation of the design rationale as this was captured in the polymorphic task hierarchy, into corresponding adaptation rules [Stephanidis et al., 1998]. A detailed description of the Unified User Interface Design method in general, and polymorphic task hierarchies in particular, can be found in [Savidis et al., 1997].

One of the key problems in the development of self-adapting user interfaces today is the inadequacy of available evaluation methods and techniques to be used for the evaluation of adaptable and adaptive interfaces. Specifically, existing evaluation methods are appropriate for assessing "static" user interfaces, but not the way and extent to which the dynamic adaptation facilities of the interface affect interaction qualities, such as accessibility, usability, acceptability, etc. Although there have been several attempts in the past to construct both objective and subjective expert- and user-based evaluation methods in the are of interface adaptation (e.g., [Totterdell & Boyle, 1990; Grüniger & van Treeck, 1993; Höök, 1997]), the lack of understanding of the dynamic dimensions of adaptive user interfaces (as well as of the differences introduced by alternative approaches to achieving and "driving" adaptive behaviour), compromises the applicability of solutions that have been suggested to date. The main deficiency of most of the aforementioned approaches is that they fail to identify the adaptation-oriented characteristics of the user interface which have detrimental, or, adversely, beneficial effects on interaction.

Due to these shortcomings, the approach taken in the evaluation of the adaptable and adaptive user interface of the AVANTI Web browser has been the introduction of a two-fold assessment process, which involved:

(a) Iterative, expert-based assessment cycles in the design of appropriate interaction styles, the definition of adaptation rules, and the development of the decision mechanism for materialising the required adaptation behaviour; expert-based assessment has been intended to compensate for the lack of appropriate evaluation techniques for adaptation-capable user interfaces and the lack of empirical evidence upon which to base the design of adaptations.

(b) End-user based evaluation activities (using questionnaires, observations and interviews), intended to assess the overall usability and accessibility of the user interface.

2. Expert evaluation

Expert evaluation activities within the development of the AVANTI Web browser aimed to employ accumulated knowledge and experience in the areas of user interface design, usability, and assistive technology, for: (a) the design of alternative interaction styles that cater for the different user and usage-context requirements, as well as (b) the design of appropriate adaptation behaviour to be built in the resulting interface.

2.1 Evaluating the design of interaction styles

Early evaluation activities were intended to assess the appropriateness of the designed interaction styles for the specific interaction context and the particular user characteristics for which they were intended. Particular emphasis was placed in the evaluation of the accessibility features provided by the designed interaction patterns to the target disabled user categories (i.e., blind and motor-impaired).

The experts reviewed each interaction style separately, based on established accessibility and usability guidelines and heuristics (e.g., [Vanderheiden & Kaine-Krolak, 1995; ISO, 1997; Microsoft, 1997; Story, 1998]), and were asked to identify potential accessibility, usability, or other problems of each style, as well as to propose possible improvements in the design, based on their experience. The outcome of these inspection activities was collected and analysed, and the results were fed back into the design process, where they have led to three types of intervention to designed interaction styles: (a) re-design of styles, based on identified problems, or on contributed ideas towards their enhancement; (b) elimination of redundant (due to the similarity in the characteristics of the end-users or the usage contexts they were intended to cater for) interaction styles; (c) introduction of new interaction styles, to cover user characteristics and contexts of use that were not addressed adequately by existing styles.

2.2 Validating the adaptation rules

The development of the adaptation rules took place in two steps. Firstly, the rules were defined by a group of experts, through several iterations following each task-decomposition phase, as well as each stage in the definition / selection of alternative interaction styles. Secondly, a process was defined, to assess the design of adaptations by validating the resulting adaptation rules. A detailed description of the adaptations rules is available in [Stephanidis et al., 1997], while a summarising account can be found in [Stephanidis et al., 1998].

The validation of the adaptation rules has itself taken place in three consecutive phases: evaluation of the rules by experts; verification of the adaptation mechanism on a per-rule basis; and, verification of the adaptation mechanism across sets of rules. The results of the experts' assessment phase have led to four types of intervention to adaptation rules: (a) elimination of rules that were deemed inappropriate, or not sufficiently supported; (b) introduction of new rules (based on the recommendations of the experts); (c) modification of the rules' triggering conditions (e.g., adding, or removing a particular user characteristic from the description of the triggering situation); (d) modification in the rules' decisions (e.g., addressing a particular situation only through guidance, instead of through guidance and extensive interim feedback).

The validation of adaptation rules was followed by the verification of the adaptation mechanism on a per-rule basis, and verification of the system's adaptation behaviour across sets of rules. The mechanism used for valuating rules and carrying out the respective adaptation decisions was tested for consistent behaviour, by: (i) testing that the triggering conditions for each individual rule led to the desired (adaptation) behaviour on the part of the user interface; (ii) testing sets of rules in combination, to assess the degree to which they affect each other from a functional, as well as from the user's point of view. The former procedure (i.e., testing rules individually) was performed by examining the defined rules one-by-one and verifying system behaviour, when the activation parameters were set, or changed. A "wizard of oz" technique was used to simulate the functionality of the User Modelling Server[12]. The later procedure (i.e., testing combinations of rules), was performed through the development of representative scenaria, where multiple activation parameters were set or changed simultaneously.

The verification procedure resulted in the identification of conflicts in the activation of specific styles and inappropriate activation of certain rules in specific tasks. The main problem arose from the redundant activation of styles under certain conditions. The outcomes of the validation procedure initiated specific modifications in the pre-defined rules, as well as the adaptation mechanism itself.

3. User-based evaluation

Formal usability evaluation studies of the AVANTI Web browser have been carried out in the context of the experimental and field evaluations of the AVANTI system in the three user trial sites in Kuusamo, Siena and Rome [Andreadis et. al., 1998]. These experiments evaluated the overall usability of the AVANTI information systems following a common evaluation framework. The trials were performed on distributed heterogeneous network environments supporting different access points, including: public information kiosks, home computers, travel agency offices, and laboratory sites. The subjects that took part in the experiments included citizens and tourists at the trial sites, as well as travel agency staff (in the case of the Siena information system). In terms of physical abilities, subjects were drawn from all three categories supported by the project, i.e., able-bodied, blind and motor-impaired. In total, there were 175 subjects in all three experiments, exposed to more than one instance of the user interface, sometimes through iterative evaluation sessions.

The usability goals set up by the common evaluation framework, and assessed during the experiments were: learnability, efficiency of use, memorability, errors, satisfaction, user attitude, adaptability and adaptivity. The experiments adopted a task-based evaluation approach, utilising both qualitative and quantitative evaluation methods. The tools which were utilised include observations and interviews (qualitative) as well as attitude scale questionnaires and log-files (quantitative). The functionalities of the AVANTI Web Browser as well as the supported adaptability and adaptivity features were addressed in the observation sessions, the interviews and the subjective evaluation (attitude scale) questionnaires.

The results of the evaluation were encouraging, both in terms of user acceptance of the characteristics of the interface, and in terms of the fulfilment of the initial goals that led to the employment of adaptation techniques in the user interface. In particular, adaptability addressing accessibility issues for the various end-user groups proved quite successful, as each user category conceived the interface as having been specifically developed to cater for their particular requirements. The results were similar for the non-disability related categories in which users were classified (e.g., according to their

[12] For this purpose, a software module that simulates the functioning of a user model server has been developed and was used for "manually" generating the dynamic user situations.

computer expertise).

Adaptivity was assessed to a lesser degree than adaptability during the evaluation, mainly due to the following reasons: (a) adaptivity requires that interactive sessions are rather lengthy, so that adequate information about the user and the context of use is collected, before any practically useful inferences can be made, and (b) existing user interface evaluation techniques do not offer themselves for the evaluation of dynamically changing, non-deterministic (from a user's perspective) systems. As far as the first of the above issues is concerned, the typical duration of the interaction sessions performed during the experimental activities was not adequate for the extraction of dependable inferences that dynamic adaptation could be based on. As a result, users were aware of only a minimal set of adaptive features in the interface; however, their reaction to the features they did observe was positive. As far as the availability of empirical methods and techniques for the evaluation of adaptivity is concerned, it has already been argued that existing knowledge in the area of user interface evaluation is inadequate for the derivation of appropriate techniques and instruments to measure the effects of adaptive system behaviour on interaction.

4. Conclusions

As already argued, there do not exist today evaluation methods and techniques that adequately address the assessment of adaptable and adaptive user interfaces, and their evaluation must, therefore, be carefully planned and conducted on a case-by-case basis. Specifically, evaluation of adaptation-capable user interfaces should aim to identify those aspects of the interface that have beneficial / detrimental effects on the accessibility and interaction quality offered by the interface for different categories of users and in different contexts of use. Two coarse evaluation dimensions can be derived from the above goal. The first concerns the appropriateness of the different instantiation styles for the purpose they were developed. This entails the assessment of the styles themselves as individual interactive artefacts and as components of the overall interface, as well as the assessment of the design rationale / decision logic that activates (or deactivates) these styles, based on user and usage characteristics. The second dimension concerns the evaluation of the dynamic adaptation (adaptivity) in the interface. This is in fact the most difficult part to evaluate, as there are multiple factors that determine the various qualities of the interface. For example, an adaptation may be conceived as entirely dissatisfactory by the user if: (a) the adaptation logic itself is flawed; (b) the "triggers" of the adaptation were wrongly inferred by the user modelling component; (c) the adaptation was not "timely" (e.g., it came "too late" from the user's perspective); or (d) the adaptation policy is not satisfactory (e.g., because the user is not given enough control over it).

To counterbalance the inherent difficulties in evaluating dynamic adaptation in the interface, evaluators should plan the evaluation process carefully from the early design phases, and should actually base the evaluation plan on the overall design process. Thus, evaluation should not be restricted to summative activities; rather, it should proceed in parallel to the design of the user interface and should strive to identify deficiencies and possible problems as early as possible, informing and guiding the development process. The evaluation activities of the AVANTI Web browser can be considered as preliminary steps towards generic methods and techniques for the evaluation of adaptation-capable user interfaces. However, a lot more research and practical experience are required in this direction, before we can derive valuable results that will be reusable across application domains, user categories and contexts of use.

5. References

[Andreadis et. al., 1998] Andreadis, A., Giannetti, L., Marchigiani, E., Rizzo, A., Schiatti, E., Tiberio, M., Pentila, M., Perala, J., Leikas, J., Suihko, T., Emiliani, P.L., Bini, A., Nill, A., Sabbione, A., Sfyrakis, M., Stary, C., & Totter, A. Global evaluation of the experiments. ACTS AC042 AVANTI Project, Deliverable DE030, August 1998.

[Fink, Kobsa & Nill, 1997] Fink, J., Kobsa, A., & Nill, A. Adaptable and Adaptive Information Access for All Users, Including the Disabled and Elderly. In A. Jameson, C. Paris, & C. Tasso (Eds.), *Proceedings of the 6th International Conference on User Modelling (UM '97)*, Sardinia, Italy (pp. 171-173). New York: Springer-Verlag, 1997.

[Fink, Kobsa & Nill, 1999] Fink, J., Kobsa, A., & Nill, A.: Adaptable and Adaptive Information Provision for All Users, Including Disabled and Elderly People. *New Review of Hypermedia and Multimedia*, in press.

[Grüniger & van Treeck, 1993] Grüniger, C., & van Treeck, W. Contributions of a Social Science Based Evaluation for Adaptive Design Projects. In M. Schneider-Hufschmidt, T. Kühme, & U. Malinowski (Eds.), *Adaptive User Interfaces: Principles and Practice* (pp. 319-330). Amsterdam, The Netherlands: Elsevier Science Publishers, 1993.

[Höök, 1997] Höök, K. Evaluating the Usability of an Adaptive Hypermedia System. In *Proceedings of the ACM International Conference on Intelligent User Interfaces (IUI '97)*, Orlando, USA (pp. 179-186). New York: ACM Press, 1997.

[ISO, 1997] International Standards Organisation. *Ergonomic Requirements for office work with visual display terminals, Part 11: Guidance on Usability (ISO 9241-11).* Draft International Standard (DIS). Geneva, Switzerland: International Standards Organisation.

[Microsoft, 1997] Microsoft Corporation. *The Microsoft Windows Guidelines for Accessible Software Design*. [On-line]. Available: http://www.microsoft.com/enable/dev/guidelines/ software.htm

[Savidis et al., 1997] Savidis, A., Paramythis, A., Akoumianakis, D., Stephanidis, C. Designing user-adapted interfaces: the unified design method for transformable interactions. In *Proceedings of the ACM Conference on Designing Interactive Systems: Processes, Methods and Techniques - DIS '97* (pp. 323-334), Amsterdam, The Netherlands, 18-20 August 1997.

[Stephanidis et al., 1997] Stephanidis, C., Karagiannidis, C., Paramythis, A., Sfyrakis, M., Stergiou, A., Savidis, A., Schreck, J., Fink, J. (February 1997) Report on Rules. ACTS AC042 AVANTI Project, Deliverable DE012, 82 pages.

[Stephanidis, Savidis & Akoumianakis, 1997] Stephanidis, C., Savidis, A., and Akoumianiakis, D. Unified Interface Development: Tools for Constructing Accessible and Usable User Interfaces. Tutorial 13, HCI International '97 Conference, San Fransisco, USA, August 1997. Available on-line at: http://www/proj/at-hci/html/tutorials.html.

[Stephanidis et al., 1998] Stephanidis, C., Paramythis, A., Sfyrakis, M., Stergiou, A., Maou, N., Leventis, A., Paparoulis, G., and Karagiannidis, C. Adaptable and Adaptive User Interfaces for Disabled Users in AVANTI Project. 5th International Conference on Intelligence in Services and Networks (IS&N '98), "Technology for Ubiquitous Telecommunications Services", Antwerp, Belgium, 25-28 May 1998.

[Story, 1998] Maximising Usability: The Principles of Universal Design. *Assistive Technology Journal*, 10 (1), 4-12.

[Totterdell & Boyle, 1990] Totterdell, P., & Boyle, E. The Evaluation of Adaptive Systems. In D. Browne, P. Totterdell, & M. Norman (Eds.), *Adaptive User Interfaces* (pp. 161-194). London: Academic Press, 1990.

[Vanderheiden & Kaine-Krolak, 1995] Vanderheiden, GC. & Kaine-Krolak, M. Access to Current and Next-Generation Information Systems by People with Disabilities [On-line]. Available at: http://www.tracecenter.org/docs/access_info_sys/full_doc.htm

Software Documentation with Animated Agents

Fiorella de Rosis, Berardina De Carolis and Sebastiano Pizzutilo

Department of Informatics, University of Bari, Italy
{derosis,decarolis,pizzutilo}@di.uniba.it
http://aos2.uniba.it:8080/IntInt.html

Abstract

We show how a formal model of interaction can be employed to generate documentation on how to use an application, in the form of an Animated Agent. The formal model we employ is XDM (Context-Sensitive Dialog Modeling), an extension of Coloured Petri Nets that enables representing user-adapted interfaces, simulating their behaviour in different contexts and making semaiutomatic pre-empirical evaluations of consistency and complexity. XDM-Agent is a personality-rich animated character that uses this formal model to illustrate the role of interface objects and to explain which tasks may be performed and how they may be performed. The behaviour of this agent is programmed by a schema-based planning (the agent's 'Mind'), followed by a surface generation (its 'Body'), in which verbal and nonverbal acts are combined appropriately. The agent's personality, that is the way its Mind is programmed and its Body appears to the user, may be adapted to the user characteristics. The potential interest of applying to software documentation the HCI metaphor of 'interacting with a friend' is discussed.

1. Introduction

Documentation is an essential component of software production that requires a considerable investment of resources and time. The result of this effort is often a partially adequate product, that must be revised every time a new release of the application is delivered and/or a new version the documentation, in a different language, has to be prepared. These problems led different research groups to see the production of documentation as strictly linked to software implementation. Documentation may refer to several aspects of software and may be addressed, consequently, to several types of users. It may be aimed at *software understanding*, that is at "the reconstruction of logic, structure and goals that were used in writing a program in order to understand what the program does and how it does it", as in MediaDoc (Erdem et al, 1998); in this case, the main users of documentation produced are software engineers. Alternatively, it may be aimed at producing a *user manual*, that describes how a given application can be used. In this case, the addressees of documentation produced are the end users of the application; the need for information of these users may vary according to the tasks they have to perform, to the frequency of use of the application, to the level of experience with the application itself and so on. The distinction between the two mentioned categories of documentation becomes more fuzzy when the software concerned is a programming language.

The idea of producing a User Manual as a byproduct of interface design and implementation becomes more practicable if a unique formal model of interaction is employed as a knowledge base to the two purposes. The most popular formal models and tools that had originally been proposed to guide interface design and implementation have been employed to automatically produce help messages; the most notable examples in this field are Petri Nets (Palanque et al, 1993) and HUMANOID Hyper Help (Moriyon et al, 1994). In these systems, the help messages are presented as texts or hypertexts, in a separate window. To complete the software documentation, animations have been proposed as well (for example, in UIDE), that combine audio, video and demonstrations to help the user to learn how to perform a task (Sukaviriya et al, 1994).

Other projects extended their goal towards the idea of generating, from a knowledge base, the main components of an instruction manual (Thimbleby and Ladkin, 1997; Novick and Tazi, 1998): DRAFTER and ISOLDE are the most significant examples in this field, their main purpose being the generation of multilingual manuals from a unique knowledge base (Paris and Vander Linden, 1996; Scott, 1996; Isolde, Web site). Some of these Projects start from an analysis of the manuals of some wellknown software products (for instance, Mac Write or Tcl-Tk), to examine which types of information these manuals include and which is the linguistic structure of each of them (Hartley and Paris, 1991). By adopting the metaphor of 'emulating the ideal of having an expert on hand to answer questions', I-Doc (an Intelligent Documentation production system) analyses the interactions that occur during expert consultations, to categorize the users' requests and to identify the strategies they employ for finding the answer to their question issues (Johnson and Erdem, 1991). This study confirms that the questions users request are a function not only of their tasks, but also of their levels of experience: more system-oriented questions are asked by novices, while experts tend to ask goal-oriented, more complex questions.

With the recent spread of research on animated characters, the idea of emulating, in a User Manual, the interaction with an expert, has a natural concretisation in implementing such a manual in the form of an Animated Agent. The most notable examples of Pedagogical Agents are Steve, Adele, Herman the Bug, Cosmo and PPP-Persona, all aimed at some form of intelligent assistance, be it presentation, navigation on the Web, tutoring or alike (Johnson, 1999; André et al, 1998; Lester at al, 1999; Rickel and Johnson, 1998). Some of these Agents combine explanation capabilities with the ability to provide a demonstration of the product, on request; the application fields to which they apply are complex systems or Web sites illustration. Several Projects at Microsoft Research apply this technology to online guide during interaction with a program (Ball et al, 1997).

In this paper, we present the first results of an ongoing Project that is aimed at generating an agent-based online User Manual from a user-adapted model of interaction. After justifying why we selected an Animated Agent as a presentation tool, we describe the formal model we employ (XDM); we then show XDM-Agent's behaviour and how it is influenced by its 'personality'; we give some details about the state of implementation of this system and conclude with some comparison with related works.

2. Why an Animated Agent

The starting point of our research on automated production of software documentation is XDM, a formalism and a tool that we employed in research and teaching for several years, to design, simulate and evaluate user-adapted interfaces (de Rosis et al, 1998). As a first step of our research on software documentation, we studied how this formalism could be employed to generate various types of hypertextual help, such as*: 'which task may I perform?'*, '*how may I* *perform this task? 'why is that interaction object inactive?'*, and others (De Carolis et al, 1998). To this aim, the formal description of interaction was read and interpreted, and schema and ATN-based natural language generation techniques were employed to produce the answers. Shifting from hypertextual helps to an Agent-based manual entails several advantages and raises several methodological problems.

2.1. The advantages

As we said, the main opportunity offered by Animated Agents is to see the software documentation as the result of a 'conversation with some expert in the field'. In messages delivered by Animated Agents, verbal and nonverbal expressions are combined appropriately to communicate information: this enables the documentation designer to select the media that is most convenient to vehicle every piece of information. As several media (speech, body gesture, face expression and text) may be presented at the same time, information may be distributed among the media and some aspects of the message may be reinforced by employing different media to express the same thing, in order to make sure that the user remembers and understands it. A typical example in the software documentation field is deixis: when helps are provided in textual form in a separate window, indicating unambiguously the interface object to which a particular explanation refers is not easy; in HUMANOID, for instance, this problem is solved by employing the same color code to denote the object, in the application and in the help window. An animated agent that can overlap to the application window may solve the same problem much more naturally, by moving towards the object, pointing at it, looking at it and referring to it by speech and/or text.
A second, main advantage, is in the possibility of demonstrating the system behaviour (after a '*How-to'* question) by mimicking the actions the user should do to perform the task and by showing the effects these actions will produce on the interface: here, again, gestures may reinforce natural language expressions, for instance by evoking abstract concepts such as the temporal relations among tasks.
A third advantage is in the possibility of making visible, in the Agent's attitude, the particular phase of dialog: by expressing 'give turn', 'take turn', 'listening', 'agreeing', 'doubt' and other meta-conversational goals (Pelachaud et al, 1999), the Agent may give the users the impression that they are never left alone in their interaction with the documentation system, that this system really listens to them, that it shows whether their question was or wasn't clear,… and so on.

2.2 The problems

Shifting from a hypermedia to an agent-based user manual corresponds to a change of the interaction metaphor that implies revising the generation method employed. In hypermedia, the main problems where to decide 'which information to introduce in every hypermedia node', 'which links to further explanations to introduce', 'which media combination to employ to vehicle a specific message', in every context and for every user. In agent-based presentations, the 'social relationship metaphor' employed requires reconsidering the same problems in different terms; it has, then, to be established 'which is the appropriate agent's behaviour' (again, in every context and for every user), 'how can the agent engage the users in a believable conversation' by providing, at every interaction turn, the 'really needed' level of help to each of them, 'how should interruptions be handled and user actions and behaviours be interpreted' so as to create the impression of interacting online with a tool that shares some of the characteristics of a human helper. These problems are common to all Animated Agents: in our project, we have examined how they may be solved in the particular case of software documentation.

3 The interface description formalism

To explain how a given application may be used, our Animated Agent employs two knowledge sources:

- a *formal description of the application interface,* in the form of a XDM-model and
- a *description of the strategies that may be employed in generating the explanation,* in the form of a Plan Operators' Library and of a Library of ATNs.

Let us, first of all, describe the first source of knowledge.

XDM (Context-Sensitive Dialog Modeling) is a formalism that extends *Coloured Petri Nets* (CPN: Van Biljon, 1988) to describe user-adapted interfaces. A XDM model includes the following components:

- a description by abstraction levels of how *tasks* may be decomposed into complex and/or elementary subtasks (with Petri Nets), with the relations among them;
- a description of the way elementary tasks may be performed (with a 'logical and physical projection' of CPN's transitions); this description consists in a set of tables that specify the task associated with every transition, the action the user should make to perform it and the interaction object concerned;
- a description of the display status before and after every task is performed (with a 'logical and physical projection' of CPN's places); this description consists in a set of tables that specify information associated with every place and display layout in every phase of interaction.

To model user-adaptation, conditions are attached to transitions and to places, to describe when and how a task may be performed and how information displayed varies in every category of users. This enables the designer to reserve access to particular tasks for particular categories of users and to vary, with the user characteristics, the way tasks are performed and the display layout appears. A detailed description of this formalism may be found in (de Rosis et al, 1998), where we show how we used it in different projects to design and simulate a system interface and to make semiautomatic evaluations of consistency and complexity.

In the production of the Animated User Manual, we employ a simplified version of XDM, in which Petri Nets are replaced with UANs (User Action Notations: Palanque et al, 1996). Like PNs, UANs describe tasks at different levels of abstraction: a UAN element represents a task; temporal relations among tasks are specified in terms of a few 'basic constructs'; given two UAN elements, the following relations may hold between them:

- sequence: A° B
- iteration: (A)*
- choice: A | B
- order independence: A & B
- concurrency: A || B
- interruptability: B → A
- interleavability: A ⇔ B

These constructs may be combined, to describe the decomposition of a task T as a string in the alphabet that includes UAN elements and relation operators.

> For example: T = (A ° (B | C))* ° D
> indicates that subtasks B and C are alternative, that A has to be performed before them, that the combination of tasks A, B and C may be iterated several times and that, finally, the task T must be concluded by the subtask D. Notice that subtasks A, B, C and D may be either elementary or complex; at the next abstraction level, every complex task will be described by a new UAN.

A UAN then provides a linearised, string-based description of the task relationships that are represented graphically in a Petri Net. Elements of UANs correspond to PNs' transitions; logical and physical projections of these elements describe how tasks may be performed; conditions attached to UANs' elements enable defining access rights. **Figure 1** shows a representation of the knowledge base that describes a generic application's interface, as an Entity-Relationship (E-R) diagram.

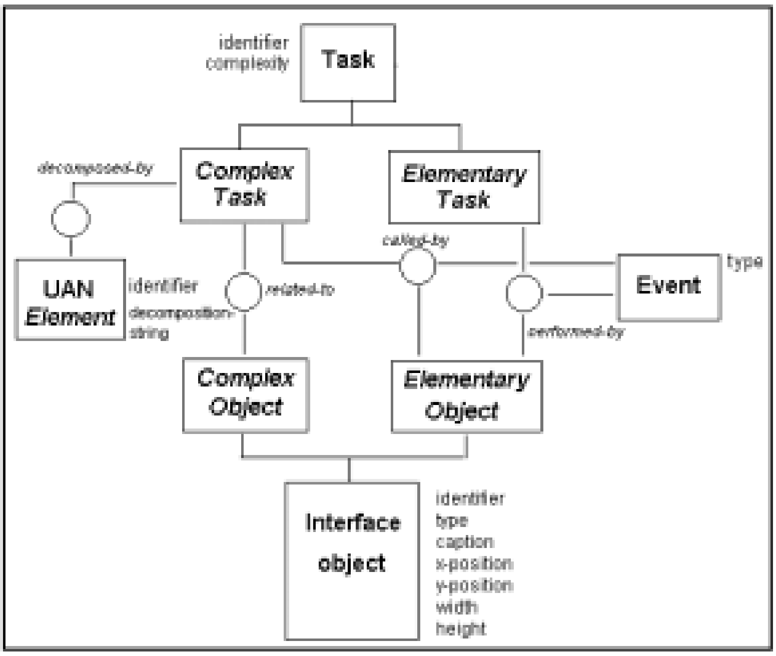


Figure 1: Entity-Relationship representation of the application-KB

UANs' elements, Tasks (either complex or elementary), Interface Objects (again, complex or elementary) and Events are the main entities. Elementary interface objects in a window may be grouped into complex objects (a toolbar, a subwindow etc). A Task is associated with a (complex or elementary) Interface Object; a UAN describes how a complex task may be decomposed into subtasks; an elementary task may be performed by a specific Event on a specific elementary Interface Object; an elementary Interface Object may open, as well, a new window that enables performing a new complex task. Adaptivity is represented, in the E-R diagram, through a set of user-related conditions attached to the entities or to the relations (we omit these conditions from Figure 1 and from the example that follows, for simplicity reasons). A condition on a task defines access rights to that task; a condition on an object defines the user category to which that object is displayed; a condition on the task-object-event relation defines how that task may be performed, for that category of users, …and so on.

Let us reconsider the example we introduced in Section 3. Let T_i be a UAN element and UAN(T_i) the string that describes how T_i may be decomposed into subtasks, in the UAN language. Let Task(T_i), Obj(T_i) and Ev(Task(T_i), Obj(T_i)) be, respectively, the task associated with T_i, the interface object and the event that enable the user to perform this task. The application-KB will include, in this example, the following items:

UANs' elements:	T, A, B, C, D
Complex Tasks:	Task (T): main database management functions
	Task (A): input identification data
Elementary Tasks:	Task (B): delete record
	Task (C): update record
	Task (D): exit from the task
UAN(T):	(A ° (B | C))* ° D
Interface Objects:	Obj (T): W1
	Obj (A): W2
	Obj (B): B1
	Obj (C): B2
	Obj (D): B3
Events:	Ev(Task(D), Obj (D)): double-click

This model denotes that database management functions (that can be performed in window W1) need first to input identification data (with window W2), followed by deleting or updating a record (buttons B1 and B2); this combination of tasks may be repeated several times. One may, finally, exit from this task by double-clicking on button B3.

4 XDM-Agent's behaviour

XDM-Agent illustrates the graphical interface of a given application starting from its main window or from the window that is displayed when the user requests the agent's help. The generation of an explanation is the result of a two-step process; in a planning phase, the presentation content is established; in a realisation phase, the plan is translated into a presentation.The *planning* algorithm thus establishes how the agent will describe the main elements that are related to the window, by reading its description in the application-KB. The presentation plan generated by this algorithm represents how the communicative goal of 'illustrating the window' may be achieved, in the form of a tree; 'primitive subgoals', that cannot be further decomposed, are attached to its leaves. Once the presentation plan for a window is ready, this is given as input to a *realisation* algorithm, that transforms it in a sequence of 'macro-behaviours'. We call 'macro-behaviour' the combination of verbal and nonverbal communicative acts that enables achieving a primitive goal in the plan.

The *list of macro-behaviours* that our agent is able to perform is application-independent but domain-dependent: that is, it is the same for any application to be documented but is tailored to the documentation task. To enable XDM-Agent to describe a user interface, we need to give it the ability to perform the following macrobehaviours:

a. *perform meta-conversational acts*: the agent must introduce itself, leave, take turn, give turn, make questions, wait for an answer;
b. *introduce-a-window* by explaining its role and its components;
c. *describe-an-object* by showing it and mentioning its type and caption (icon, toolbar or other);
d. *explain-a-task* by mentioning its name;

e. *enable-performing-an-elementary-task* by describing the associated event;
f. *demonstrate-a-task* by showing an example of how the task may be performed;
g. *describe-a-task-decomposition* by illustrating the relationships among its subtasks.

A macrobehaviour is obtained by combining verbal and nonverbal acts as follows:

- *verbal acts* are produced with natural language generation functions, that fill context-dependent templates with values from the application-KB; the produced texts are subsequently transformed into 'speech' or 'write a text in a balloon';
- *nonverbal acts* are 'micro behaviours' that are produced from MS-Agent's[1] animations, with the aid of a set of auxiliary functions. The list of micro behaviours that are needed in our animated user manual is shown in **Figure 2.** This list includes: (i) *object-referring gestures*: the agent may move towards an interface object or location, point at it and look at it; (ii) *iconic gestures*: the agent may evoke the relationship among subtasks, that is a sequence, a iteration, a choice, an order independence, a concurrency and so on; it can mimic, as well, the actions the user should do to perform some tasks: click, double click, keyboard entry,…and so on; (iv) *user-directed gestures*: the agent may look at the user, get closer to him or her by increasing its dimension, show a questioning or listening attitude, manifest its intention to give or take the turn, open and close the conversation with the user by introducing itself or saying goodbye.

---

a. *Greetings*
   - Self_Introduction
   - Leave
b. *Meta-Conversational-Gestures*
   - Take_Turn
   - Give_Turn
   - Questioning
   - Listening
c. *Locomotive-Gestures*
   - Move_To_Object (Oi)
   - Move_To_Location (x, y)
d. *Deictic-Gestures*
   - Point_At_Location (x,y)
   - Point_At_Object (Oi)
   - Point-At_Area ((xi, yi), (xj, yj))
e. *Relation-Evoking*
   - Evoke_Sequence
   - Evoke_Iteration
   - Evoke_Choice
   - Evoke_Order_Independence
   - ……
f. *Event-Mimicking*
   - Mimic_Click
   - Mimic_Double_Click
   - Mimic_Keyboard_Entry
   - ……
g. *Looking*
   - Look_At_User
   - Look_At_Location (x,y)
   - Look_At_Object (Oi)
   - Look_At_Area ((xi, yi), (xj, yj))
h. *Approaching-the-user*

Figure 2: Library of XDM-Agent's 'micro behaviours'

[1] MS-Agent is a downloadable software component that displays an animated character on top of an application window and enables it to talk and recognise the user speech. A character may be programmed by a language that includes a list of 'animations' (body and face gestures): these animations are the building blocks of XDM-Agent.

As we will further discuss in Section 6, the way these micro behaviours are implemented strongly depends on the animations that are included in the software we employ: in particular, a limited overlapping of gestures can be made in MS-Agent, which only enables overlapping speech and text to body animations. We therefore overlap verbal acts to nonverbal ones so that the Agent can move, speech and write something on a ballon at the same time, while we sequence nonverbal acts so as to produce ‘natural’ behaviours. As we mentioned in Section 2, we employ nonverbal acts to reinforce the message vehicled by verbal acts. So, the agent’s speech corresponds to a self-standing explanation, that might be translated into a written manual; text balloons mention only the ‘key’ words in the speech, on which users should focus their attention; gestures have the role of supporting the communication tasks that could not be effectively achieved with speech (for instance, deixis), of reinforcing concepts that users should not forget (for instance, task relations), of supporting the description of the way actions should be done (for instance, by mimicking events) and, finally, of giving the users a constant idea of ‘where they are, in the interactive explanation process (for instance, by taking a ‘listening’ or ‘questioning’ expression). Finally, like for all embodied characters, speech and gestures are employed, in general, to make interaction with the agent more ‘pleasant’ and to give users the illusion of ‘interacting with a companion’ rather than ‘manipulating a tool’.

5. XDM-Agent’s “personality”

A typical software manual includes three sections[2]:

- a *tutorial*, with exercises for new users,
- a series of step by step *instructions for the major tasks* to be accomplished and
- a ready-reference *summary of commands*.

To follow the ‘minimal manual’ principle (“the smaller the manual, the better”: J M Carrol, cited in Addison and Thimbleby, 1994), the Agent should start from one of these components, provide the “really needed minimal” and give more details only on the user’s request. The component from which to start and the information to provide initially may be fixed and general or may be varied according to the user and to the context. In the second case, as we mentioned in the Introduction, the user goals, his/her level of experience and his/her preference concerning the interaction style may drive selection of the Agent’s “explanation attitude”. Embodiment of the Agent may be a resource to make this attitude explicit to the user, by varying the Agent’s appearance, gesture, sentence wording and so on.

XDM-Agent is able to apply two different approaches to interface description; in the task-oriented approach, it systematically instructs the user on the tasks the window enables performing, how they may be performed and in which sequence and provides, if required, a demonstration of how a complex task may be performed. In command-oriented descriptions, the agent lists the objects included in the window in the order in which they are arranged in the display and provides a minimal description of the task they allow to perform; other details are given only on the User request. Therefore, in the first approach the Agent takes the initiative and provides a detailed explanation, while in the second one the initiative is ‘mixed’ (partly of

[2] This list of contents was found by Hartley and Paris (1996) in their analysis of a MacWrite manual, and we found the same overall structure in our analysis of the Visual Basic’s hyper-manual.

the Agent, partly of the User), explanations are, initially, much less detailed and a dialog with the User is established, to decide how to go on in the description of the application.
If the metaphor of 'social interaction' is applied to the User-Agent relationship, the two approaches to explanation can be seen as the manifestation of two different 'help personalities' in the Agent (de Rosis and Castelfranchi, 1999):

- a *overhelper*, that tends to interpret the implicit delegation received by the user in broad terms and explains anything he presumes the user desires to know, and
- a *literal helper*, that provides a minimal description of the concepts the user explicitly asks to know[3].

These help attitudes may be seen as particular values of the dominant/submissive dimension of interpersonal behaviour[4], which is considered to be the most important factor affecting human-computer interaction (Nass et al, 1995). Although some authors proved that dominance may be operationalised by only manipulating the phrasing of the texts shown in the interface and the interaction order (again, Nass et al, 1995; but also Dryer, 1996), others claim that the user appreciation of the interface personality may be enhanced by varying, as well, the Agent's 'external appearance': body posture, arm, head and hands gestures, moving (Isbister and Nass, 1998; Arafa et al, 1998; Bell and Breese, 1998). By drawing on the cited experiences, we decided to embody the overhelping, dominant attitude in a more 'extroverted' agent that tends to employ a direct and confident phrasing and to gesture and move much. We embody, on the contrary, the literal helper, submissive attitude in a more 'introverted' agent, that employs lighter linguistic expressions and moves and gestures less. In order to enhance matching of the Agent's appearance with its underlying personality, we select *Genie* to represent the more extroverted, dominant personality and *Robby* to represent the submissive one[5]: this is due in part to the way the two characters are designed and animated, in the MS-Agent tool, and in part to the expectation they raise in the user: Genie is seen as more 'empathetic', someone who takes charge of the Users and anticipates their needs, while Robby is seen as more 'formal', someone who is there only to respond to orders. **Figure 3** summarises the main differences between the two characters.

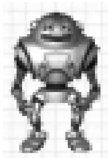
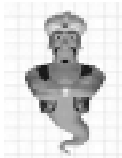
How do we combine the Agent's personality with the User's characteristics? Some authors claim that task-based explanations would be more suited to novice users and object-oriented explanations more suited to experts. For instance, empirical analysis of a corpus of documents, in TAILOR, showed that complex devices are described mainly in an object-oriented way in adult encyclopedias, while descriptions in junior encyclopedias tend to be organized in a process-oriented, functional way (Paris, 1989). On the other side, a 'dominant', 'extroverted' personality is probably more suited to a novice, while a 'submissive' and 'formal' one will be more easily accepted by an expert user. This led us to select *Robby* for novices and *Genie* for experts.

---

3 We give a more detailed definition of these 'dominance' aspects of personality in (de Rosis and Castelfranchi, 1999), where we also define, more in general, the various attitudes towards delegation and help that may be established in multiagent systems.

4 According to Nass et al, 1995, dominance is marked, in general, by a behaviour that is "self-confident, leading, self-assertive, strong and take-charge"; submissiveness is marked, on the contrary, by a behaviour that is "self-doubting, weak, passive, following and obedient".

5 Genie and Roby are Microsoft-Agent's characters.

Robby	Genie
Object-oriented presentation	*Task-oriented presentation*
submissive	dominant
introverted	extroverted
is rather 'passive'; says the minimum and waits for the user's orders	is very 'active': takes the initiative and provides detailed explanations
employs 'light' linguistic expressions, with indirect and uncertain phrasing (suggestions)	employs 'strong' linguistic expressions, with direct and confident phrasing (commands)
gestures the minimum: minumum locomotion, limited movements of arms and body avoids getting close to the user	gestures are more 'expansive': more locomotion, wider movements, gets closer to the user
speaks slow	speaks high

Figure 3: personality traits of Robby and Genie,and corresponding differences in behaviours

6. Implementation

We implemented XDM-Agent in Java and Visual Basic, under Windows95. Adaptation of the generated documentation to the agent's personality is made at both the planning and the realisation phase. In planning, a task-oriented and more detailed or object-oriented and less detailed discourse plan is produced for the two agents. In surface generation, the plan leaves are translated into different realisations of macro-behaviours.

At the planning level, adaptation is made by introducing personality-related conditions in the planning schemas. At the surface realisation level, a unique Behaviour Library is employed in the two cases, with different ways of realising every behaviour. A representation of two portions of plan for Genie and Robby are given in **Figure 4**; this figure shows the difference between a task-oriented and an object-oriented approach to the interface presentation; it shows, as well, that the two plan trees have some leaves in common.
In the task-oriented plan, a window is introduced by mentioning the complex task that this window enables performing; the way this complex task is decomposed into less complex subtasks is then described, by examining (in the application-KB) the UAN associated with this window. For each element of the UAN, the task and the associated object are illustrated; if the task is 'elementary', the event enabling to perform it is mentioned as well; if the task is complex, the user is informed that a demonstration of how to perform it may be provided, if requested. Task relations (again, from the UAN) are then illustrated. Description goes on by selecting the next window to describe, as one of those that can be opened from the present window. In the object-oriented plan, interface objects are described by exploring their hierarchy in a top-down way; for each elementary object, the associated task is mentioned. The turn is then given to the user, who may indicate whether and how to proceed in the explanation.

Figure 4: a portion of the task-oriented and of the object-oriented plans

Planning is totally separated from realisation: we employ, to denote this feature, the metaphor of 'separating XDM-Agent's *Mind* from its *Body'*; we might associate, in principle, Robby's appearance and behaviour to a task-oriented plan (that is, Robby's Body to Genie's Mind) and the inverse. This separation of the agent's body from its mind gives us the opportunity to implement the two components on the server-side and on the client-side respectively, and to select the agent's appearance that is preferable in any given circumstance. An example of the difference in the behaviour of the two agent's personalities is shown in **Figure 5, a** and **b.**

Genie

Macro-Behaviour	***Speech***	***Text balloon***	***Gesture***
Introduce-a-window (task-oriented)	In this window, you can perform the main database management functions:		point-at-area look-at-user
Explain-task Describe-object	- to select a database management function, use the commands in this toolbar;	Select database management functions	move-to-object point-at-object look-at-object look-at-user
idem	- to input identification data, use the textfields in this subwindow;	Input identification data	idem
	…….	…….	…….
Introduce-complex-task	There are two database management functions you may select:		
Explain-task Describe-object Enable-performing-a-task	- to update an existing record, click on the 'Update' button;	Update	move-to-object point-at-object look-at-object look-at-user mimic-click
idem	- to delete a record, click on the 'Delete' button;	Delete	idem
	…….	…….	…….
Describe-task-relation	These tasks are independent of each other; you must choose one of them and perform it.	Choose one	approach-the-user evoke-choice

Figure 5 a: a portion of task-oriented description of a window, by Genie

Robby

Macro-Behaviour	***Speech***	***Text balloon***	***Gesture***
Introduce-a-window (object-oriented)	I'm ready to illustrate you the objects in this window:		look-at-user
Introduce-complex-object	- a toolbar, with 5 buttons:		
Describe-object Explain-task	the first one enables you to update an existing record	Update	point-at-object look-at-object
idem	the second one enables you to delete a record	Delete	idem
	…..		

Figure 5 b: a portion of an object-oriented description of a window, by Robby

7. Interest and limits of our system

The first positive result of this research is that we could verify that the formal model we employed to design and evaluate the interface (XDM) enables us, as well, to generate the basic components of an Animated Instruction Manual. Planning the structure of the presentation directly from this formal model contributes to insure that “the manual reflects accurately the system’s program and that it can be viewed as a set of pre-planned instructions” (Addison and Thimbleby, 1994). An Animated User Manual such as the one we generate is probably suited to the needs of ‘novice’ users; however, we are not sure that the more complex questions an expert makes can be handled efficiently with our application-KB. We plan to carefully check these problems in an evaluation study, from which we expect some hints on how to refine our system. In this study, we plan to assess which is the best matching between the two XDM-Agent’s personalities and the users characteristics, including their experience and their personality: in fact, the evidence on whether complementarity or similarity-attraction holds between the system and the user personalities is rather controversial so far (Isbister and Nass, 1999), and we suspect that no general rule can be established in this domain, and that the decision depends on the particular personality traits considered.

The present prototype of XDM-Agent has several limits: some of them are due to the generation method we employ, others to the tool:

- acthe and main gaze limit expressions generation. Some method is that we do not yet handle *interruptions*: users may take the initiative to make questions to the system only when the agent gives them the floor; ‘difficult’ task); we cannot employ ‘backchanneling
- the second main limit originates from available animations. In MS-Agent, the repertoire of gestures is rather limited, especially considering f
  examples: we cannot give an intonation to our Agents’ speech; we cannot emphasize the difference in the way they phrase their utterances (a suggestion vs a command) with a difference in ‘performative eyes’; we cannot use ‘adjectival eyes’ to express some concept’s property (for instance, a ‘di
  eyes’ to provide a feedback to the user, as in the expression of doubt,… and so on [6]. In addition, the difficulty of overlapping animations in MS-Agent does not allow us to translate into face or body gestures the higher parts of the discourse plan. XDM-Agent thus lacks of those gestures that aid in integrating adjacent discourse spans into higher order groupings (Kendon, 1972), for instance by expressing rhetorical relations among

[6] For a definition of the semantic of the various types of gaze that are employed in human and/or agent conversations, see (Pélachaud et al, 1999).

high-level portions of the plan; an example: a 'contrast' between the ways two tasks can be performed. To overcome these limits, we should build our own character, with the mentioned animations.

These limits in our agent's behaviour having been considered, we have, still, to assess whether Animated Agents really contribute to make software documentation more usable, in which conditions and for which user categories. This consideration applies to the majority of research projects on Animated Agents, which has been driven, so far, by an optimistic attitude rather than a careful assessment of the validity of results obtained (with a few exception, as in (André et al, 1998)).

8. Related work

Our research lies in the crossroad of several research areas: formal models of HCI, user adaptation and believable agents. There are, we believe, some new ideas in the way these areas are integrated into XDM-Agent: we showed, in previous papers, that a unique formal model of HCI can be employed to unify several steps of the interface design and implementation process: after analysis of user requirements has been completed, these requirements can be transferred into a UI specification model that can be subsequently employed to implement the interface, to simulate its behaviour in several contexts and to make pre-emipirical usab evaluations. In this paper, we show that the same model can be employed, as well, to produce an online user manual. Any change in the UI design must be transfered into a change in the formal model and automatically produces a new version of the interface, of the simulation of its behaviour, of usability measures and of the documentation produced.

Adaptation to the user and to the context is represented through parameters in the model and reflects into a user-adapted documentation. In particular, adaptation to the user needs about documentation is performed through the metaphor of 'changing the personality of the character who guides the user in examining the application'. That computer interfaces have a personality was already probed by Nass and colleagues in their famous studies in which they applied to computers the same theories and methods originally developed in the psychological literature for human beings (see, f.i., Nass and colleagues, 1995). Taylor and colleagues' experiment demonstrates that a number of personality traits (in the 'Five-Factor' model) can be effectively portrayed using either voice alone or in combination with appropriately designed animated characters" (Taylor et al, 1998). Results of the studies by these groups oriented us in the definition of the verbal style and the nonverbal behaviours that characterise XDM-Agent's personalities (see, f.i., Isbister and Nass, 1999); Walker and colleagues research on factors affecting the linguistic style guided us, as well, in the diversification of the speech attitudes (Walker et al, 1997). S

of their Animated Agents' personality traits (for instance: Ball and Breese, 1999; Arafa et al, 1999). We contributed, in the past, to the debate about the personality traits that might be relevant in HCI by formalising the cooperation levels and types and the way they may combine, in terms of 'personality traits' (Castelfranchi et al, 1998; de Rosis and Castelfranchi, 1999). Robby and Genie are programmed according to some of these traits; other traits (such as a critical helper, a supplier and so on) might be attributed to different characters, with different behaviours. The long-term goal of our research is to envisage a human-computer interface in which the users can settle, either implicitly or explicitly, the 'helping attitude' they need in every application: XDM-Agent is a first step towards this direction.

Acknowledgements

We thank Alessandro Assab and Floriano Cancelli for contributing to implement a prototype of XDM-Agent in the scope of their dissertations in computer science.

References

M Addison and H Thimbleby: Manuals as structured programs. *Proceedings of HCI*, Cambridge University Press, 1994. 1999

E André, T Rist and J Mueller: Integrating reactive and scripted behaviours in a life-like presentation agent. In K P Sycara and M Woolridge (Eds): *Proceedings of the Second International Conference on Autonomous Agents*. ACM Press, 1998. 1997.

J Arafa, P Charlton, A Mamdani and P Fehin: Designing and building Personal Service Assistants with personality. *Proceedings of the Workshop on Embodied Conversational Characters*, S Prevost and E Churchill (Eds), Tahoe City,

G Ball, D Ling, D Kurlander, J Miller, D Pugh, T Skelly, A Stankosky, D Thiel, M van Dantzich and T Wax: Lifelike computer characters: the Persona project ar Microsoft. In J M Bradshaw (Ed): *Software Agents.* AAAI/MIT Press, Menlo Park,

G Ball and J Breese: Emotion and personality in a conversational character. *Proceedings of the Workshop on Embodied Conversational Characters*, S Prevost and E Churchill (Eds), Tahoe City, 1999.

C Castelfranchi, F de Rosis, R Falcone and S Pizzutilo: Personality traits and social attitudes in multiagent cooperation. *Applied Artificial Intelligence*, 12, 1998.

B De Carolis, F de Rosis and S Pizzutilo: Dynamic generation of on-line help messages from formal dialogue models. DIB-Internal Report, 1998.

F de Rosis, S Pizzutilo and B De Carolis: Formal description and evaluation of user-adapted interfaces. *International Journal of Human-Computer Studies*, 1998.

F de Rosis and C Castelfranchi: How can personality factors contribute to make agents more 'believable'? *Proceedings of the Workshop on 'Behaviour Planning for Life-like Characters and Avatars'*, Sitges, 1999.

D C Dryer: Dominance and valence: a two-factor model for emotion in HCI. *AAAI Press, 1998.*

A Erdem, W L Johnson and S Marsella: Task-oriented software understanding. *Proceedings of the Automated Software Engineering Conference*, 1998. 1999

A Hartley and C Paris:Two sources of control over the generation of software instructions. *Proceedings of ACL 96*, 1996.

C Isbister and C Nass: Personality in conversational characters: building better digital interaction pertners using knowledge about human personality preferences and perceptions. *Proceedings of the Workshop on Embodied Conversational Characters*, S Prevost and E Churchill (Eds), Tahoe City,

Isolde Web site: http://www.syd.dit.csiro.au/staff/cecile/isolde.html

W L Johnson and A Erdem: Interactive explanations of software systems. *Automated Software Engineering, an International Journal.* 1996

W L Johnson: Pedagogical Agents.
http://www.isi.edu/isd/carte/ped_agents/pedagogical_agents.html

A Kendon: Some relationships between body motion and speech. In: *Studies in dyadic communication.* A W Siegman and B Pope (Eds), Pergamon, 1972.

J C Lester, J L Voerman, S G Towns and C B Callaway: Deictic believability: coordinated gesture, locomotion and speech in lifelike pedagogical agents. *Applied Artificial Intelligence*, 13 (4-5), 1999.

R Moriyon, P Szekely and R Neches: Automatic generation of help from interface design models. *Proceedings of CHU94*, Boston, 1994.

C Nass, Y Moon, B J Fogg, B Reeves and D C Dryer: Can computer personalities be human personalities? *Int J Human-Computer Studies*, 43, 1995.

D G Novick and S Tazi: Applying the Act-Function-Phase Model to Aviation Documentation. *Proceedings of SIGDOC 98*, Quebec, 1998.

P A Palanque, R Bastide and L Douarte: Contextual help for free with formal dialogue design. *Proceedings of HCI93*, Orlando, 1993.

P Palanque, R Bastide and V Sengès: Validating interactive system design through the verification of formal task and system models. In *Engineering for Human-Computer Interaction*. L J Bass and C Unger (Eds), Chapman and Hall, 1996.

C L Paris: The use of explicit User Models in a generation system for tailoring answers to the user's level of expertise. In A Kobsa and W Wahlster (Eds): *User Models in Dialog Systems*. Springer Verlag, 1989.

C Paris and K Vander Linden: DRAFTER: an interactive support tool for writing mult instructions. *IEEE Computer*, 29, 7, 1996 lingual software documentation.

C Pélachaud, I Poggi and F de Rosis: Study and generation od coordinated linguistic and gaze communicative acts. *Proceedings of AI*IA 99*, 1999.

J Rickel and W L Johnson: Task-oriented dialogs with animated agents in virtual reality. In S Prevost and E Churchill (Eds): *Workshop on Embodied Conversational Characters,* 1998.

D Scott: Computer support for authoring mult *ITRI-96-14, Information Technology Research Institute Technical Report Series*, 1996.

P Sukaviriya, J Muthukumarasamy, A Spaans and H J J de Graaf: Automatic generation of textual, audio and animated help in UIDE. *Proceedings of the Conference on Advanced Visual Interfaces*, Bari, 1994.

I C Taylor, F R McInness, S Love, J C Foster and M A Jack: Providing animated characters with designated personality profiles. *Proceedings of the Workshop on Embodied Conversational Characters*, S Prevost and E Churchill (Eds), Tahoe City, 1999.

A Thimbleby and P Ladkin: From Logic to manuals. *Proceedings of the IEEE Conference on Software Engineering*, 1997.

V R Van Biljon: Extending Petri Nets for specifying man-machine interaction. *International Journal of Man-Machine Studies*, 28, 1988.

M A Walker, J E Cahn and S J Whittaker. Improvising linguistic style: social and affective bases for agent personality. *Autonomous Agents 97*

KNOWLEDGE-BASED AND LAYOUT– DRIVEN ADAPTIVE INFORMATION PRESENTATIONS ON THE WORLD WIDE WEB

George Vouros
Dept. of Mathematics
University of the Aegean
Samos, Karlovassi
Greece

Abstract

The paper presents on-going research towards generic tools and methods for fulfilling the combined needs of information producers and consumers. It presents the principles, framework and key issues of our research, and focuses on developments concerning INFO-PRESENTER. This is an interactive system that provides knowledge-based and layout-driven information presentations, intending to satisfy the idiosyncrasies of information consumers, supporting a great amount of tailoring information to their needs, interests, preferences and background knowledge. The paper focuses mostly on themes concerning knowledge representation and layout specifications. The representation framework provides a general model for the specification of information requirements in terms of media-independent information item categories and types of information views for realising information. Layout specifications provide generic rules for tailoring information delivery, in conjunction with the information specifications. To facilitate user-tailored information production, information and layout specifications should be structured and encoded so as to facilitate their cost-effective creation and maintenance, in relation to their effective utilization for user-tailored information presentations.

1. Introduction

Critical issues concerning the use of information on the World Wide Web are the efficiency and efficacy of its creation, maintenance and exploitation. Major categories of information users are information producers and information consumers.

Information producers aim to provide information that achieves their communicative goals, addresses the needs, interests and satisfy the preferences of their intended audience. Information consumers need to tailor information to their background knowledge, preferences and abilities, and satisfy their information needs in various contexts of information use.

This paper presents on-going research towards generic tools and methods for fulfilling the above-mentioned needs of information users. It presents the principles, framework and key issues of our research, and focuses on developments concerning INFO-PRESENTER. This is an interactive system that provides knowledge-based and layout-driven information presentations. The system aims to satisfy the idiosyncrasies of information consumers and supports a great amount of tailoring information to their needs, interests, preferences and background knowledge. The paper presents system's overall architecture, and focuses on the knowledge representation framework that the system exploits, as well as on the way layouts, which drive information presentations, are specified. The representation framework provides a general model for specifying information. Information comprises domain concepts, media-independent information items related to domain aspects, and information views for realizing information items using specific media formats, subject to the contextual constraints of information use. Layout specifications provide generic rules for tailoring information delivery, in conjunction with the information specifications.

The paper is structured as follows: Section 2 presents related work towards our research goals, presents the motivations and key issues of our research. Section 3 presents the system architecture and section 4 describes the current status of system implementation and future developments. Section 5 provides concluding remarks.

2. Previous work - Motivation – Key issues

Adaptive hypermedia systems aim to tailor the content, presentation and hypermedia structure of information to the needs, interests, preferences, abilities and background knowledge of their users.

- *User needs* include information seeking goals and the real-world tasks that comprise the context in which information is consumed. For example, in case the user is operating with a device and does not know how to perform a particular task with it, then he forms the desire "to acquire knowledge about performing this task". This desire represents his need. Moreover, physical abilities and background knowledge of users may help to determine their information needs. For instance, the information needs of an expert in a domain are different from the needs of a novice. Needs determine the information interests of users.
- *Information interests* comprise subject related knowledge of the information items that users need to consume. For instance, in case the need of the user is "getting information about setting the volume of tone ringing volume' in a telephone device, then his interests are, among others, about "tone ringing" and "volume". These may specialize a generic interest, or be part of an inclusive one, recognized earlier through interaction.
- *Preferences* are about certain information modalities for communicating information, such as text over graphics and images, audio over text etc.
- *Abilities* concern physical abilities of the user, as well as network bandwidth, hardware and software performance characteristics.
- *Background knowledge* of users denotes their experience in the subject domain.

In the last few years, a lot of effort has been devoted to the development of adaptive hypermedia systems, with major emphasis to adaptive presentation of information content and adaptive navigation support. More specific types of adaptive hypermedia systems, such as adaptive educational systems, address also issues concerning adaptive sequencing of information, problem solving support and intelligent solution analysis [Spe99, Bra98, Bru98, Esp96].

Adaptive hypermedia systems utilize a conceptual, or content, model of the domain and use concepts to either tag segments of canned information, or to generate natural language descriptions of the information content, and sometimes, to generate graphical descriptions of domain knowledge or of the way information is structured. Although it seems that a unifying framework for information specification can exist, most of these systems do not propose the utilization of such a representation framework.

M.Zhou and S.Feiner [Zhou98] proposed an extensible, comprehensive and general representation framework with the aim to represent all basic visual forms (visual media formats) needed for automated graphics generation. One or more primitive visual forms, referred as object's senses (visual words), represent a domain aspect. A sense includes the necessary characteristics for specifying the role an information item may play in a presentation, as well as its form. The framework does not distinguish between medium-independent information items and medium-dependent forms for realising these items. For the presentation of an object, a sense selection mechanism selects a visual word by taking into account presentation goals and contextual constraints. However, there are cases that a presentation needs two or more pieces of information of the same domain aspect, each piece presented in multiple media and forms.

C.Welty [Wel96] proposed an ontology, which is broken into three parts: Domain aspects, medium-independent information items and information views that express information items. However, this framework does not deal with representing the necessary knowledge for choosing information items and views in the appropriate contexts.

Generally, selection of information during a presentation involves selecting the information content (i.e. media-independent items of information for achieving presentation and communicative goals), and designing the presentation by selecting the appropriate media and formats (i.e. media and form dependent views of information items) for information realization. These tasks involve reasoning about the domain, the presentation goals, the presentation context, and user characteristics. Selection of information items may be driven by the availability of views and vice-versa. Making the distinction between information items and medium-dependent views of information, makes the representation more comprehensive and supports the identification of generic information categories and types of information views. Information categories help in categorizing fragments of information according to their role in achieving presentation intentions, and types of views provide the generic forms of realizing information subject to contextual (pragmatic) constraints.

Furthermore, most of adaptive hypermedia systems do not deal with the costs of creating knowledge bases and do not emphasize on methods and tools for their efficient development. As pointed in [Par96], "… systems assume that an underlying knowledge base containing the necessary information is available or can be easily obtained". Moreover, as it is documented in [Rei94], a major obstacle towards automatic generation of technical documentation is the increased cost for knowledge base development and maintenance. This, together with the needs of reducing the costs of quality assurance, and of achieving a great amount of adaptability of information, point to the necessity of keeping information producers into the loop.

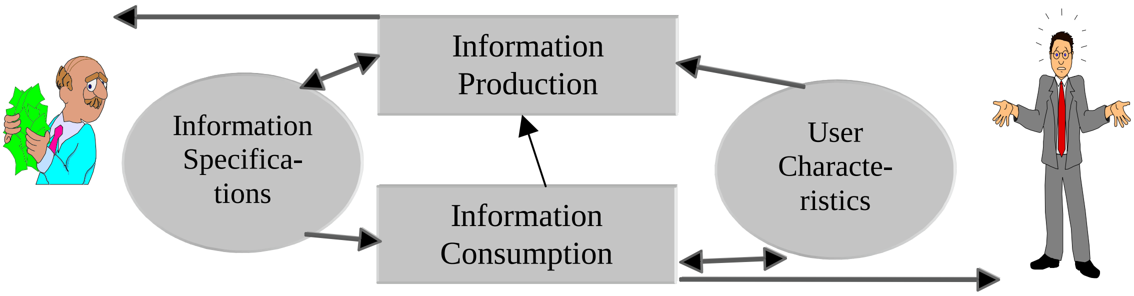


***Figure 1**: Keeping information producers and information consumers in the loop.*

Information producers need tools for defining the conceptual structure of a domain, for specifying information items about domain aspects, and for providing alternative ways of realizing information in particular contexts/settings. Furthermore, information producers must relate information to the characteristics of their intended audience. These characteristics include the needs, interests, preferences, abilities and background knowledge of potential information consumers.

Concerning information consumption, adaptivity of information must be based on specifications about the proper use of information, be driven by the on-going dialogue between user and system, by user characteristics acquired during interaction, as well as by specifications about presentations overall structure and layout.

Figure 1 describes the information production-consumption loop in which characteristics of information consumers drive information production as well.

The aim of the research reported in this paper is in empowering information producers and information consumers by providing them with the appropriate tools.

Key issues in this research are:

- The use of a general, extensible and comprehensive framework for the representation of knowledge concerning the use of multimedia information in the context of the presentation design task. The framework must be general by providing the building blocks for specifying generic information categories and types of information views. Furthermore, it must be extensible by facilitating the specification of new information categories and information types, and by facilitating the specification and proper categorization of information items and information views, according to their role in achieving presentation intentions and their use in various information-related human activities. The objective is to facilitate efficient knowledge base development and maintenance, in conjunction with effective and efficient utilisation of knowledge for information presentation. As already pointed in [Vou99a, Rei95], to reduce the cost of creating knowledge bases, and in order to facilitate their development by information producers, we sacrifice the benefits of using low-level generation techniques. Therefore, the proposed representation framework is hybrid, mixing proper knowledge structures with canned fragments of media objects. The representation framework guides information production and captures the necessary knowledge for supporting a great amount of designing user-tailored information presentations.

- The design and development of a tool that will assist and facilitate specification of information using the above mentioned representation framework. Users must specify information items that they intent to present, relate these items to the domain aspects, specify the communicative functions of these items, specify alternative forms of realizing information and associate information with user-oriented data. This tool must also assist users to the production of presentations' layout specifications.

- The design and development of a tool that will tailor and present information to consumers' requirements and characteristics. The tool must consult presentation layout specifications and exploit knowledge concerning the proper use and combination of information items. The tool must exploit the specifications of information producers.

- The use of a user-modelling framework for representing, inferring and maintaining knowledge about users by elaborating on their stated characteristics (e.g. background knowledge, abilities, software-hardware platforms, network connection), on their interaction with the system (e.g. follow-up questions, preferences in media, interests related to domain aspects etc), and on the on-going dialogue with the system (e.g. plans, goals, intentions, knowledge acquired, generic information interests about the domain).

The paper presents INFO-PRESENTER, which is an interactive prototype system that provides user-tailored information presentations. To tailor information to users, INFO-PRESENTER exploits information specifications encoded in the above mentioned representation framework, layout specifications encoded in extended HTML, the stated user characteristics, and the dialogue context with the user.

3. INFO-PRESENTER Architecture

The overall system architecture of INFO-PRESENTER, in terms of the Standard Reference Model for Intelligent Multimedia Presentation Systems [SRM98], is given in Figure 2.

Key issues related to system implementation are

(a) The framework for representing knowledge about the domain and the media objects utilized during presentation, and

(b) The layout specifications that drive information presentation.

Knowledge Base.

The knowledge base is built using the representation framework proposed in [Vou99a]. It represents domain knowledge in combination with knowledge concerning the use of multimedia information in the context of achieving communicative goals. As already pointed out, emphasis has been given to the efficiency of developing and maintaining knowledge bases, as well as to the efficient use of knowledge for constructing multimedia presentations.

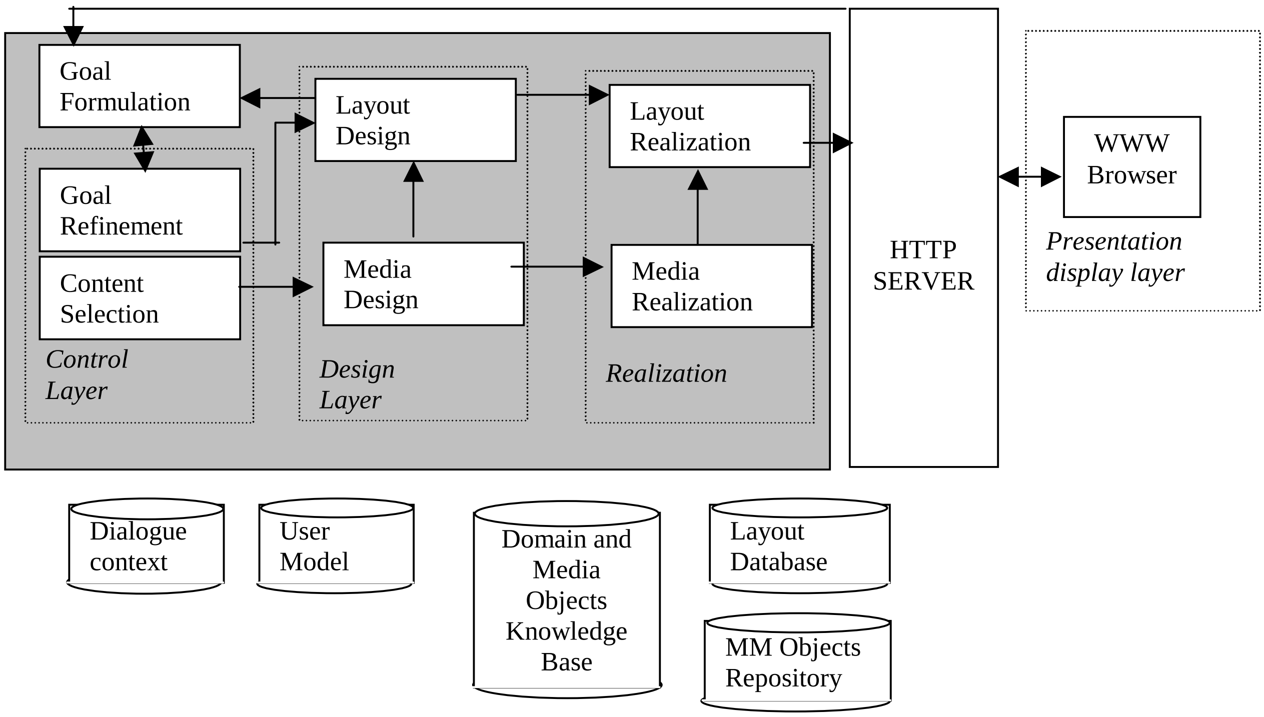


Figure 2: *INFO-PRESENTER architecture*

The representation framework

- Is extensible, generic, and makes specific ontological commitments for guiding information specification and structuring. The representation is comprehensive to represent and handle information items of domain aspects, as well as to handle multiple and heterogeneous views of these items. The representation provides the means for expressing the necessary constraints for selecting information items of domain aspects and views of these items during design of information presentations.

- Is hybrid and intermix formal knowledge structures with canned information fragments. The level in which primitive information fragments are specified and therefore, the balance between proper knowledge base structures and canned media objects is an issue to which the framework makes no commitment.

- Supports subject-based classification of the represented information. Representation of subjects and subject based classification is important for indexing information, for keeping track of user interests and, for making information accessible in the appropriate contexts (e.g. for answering follow-up questions).

The framework has been implemented using the BACK Description Logic [BACK]. Description Logics have been chosen due to their formality, and due to the reasoning facilities provided.

The representation framework is depicted in Figure 3. Each oval in this diagram represents a concept hierarchy. Based on this framework, Figure 4 depicts specification of information concerning a part of a telephone device: SET KEY.

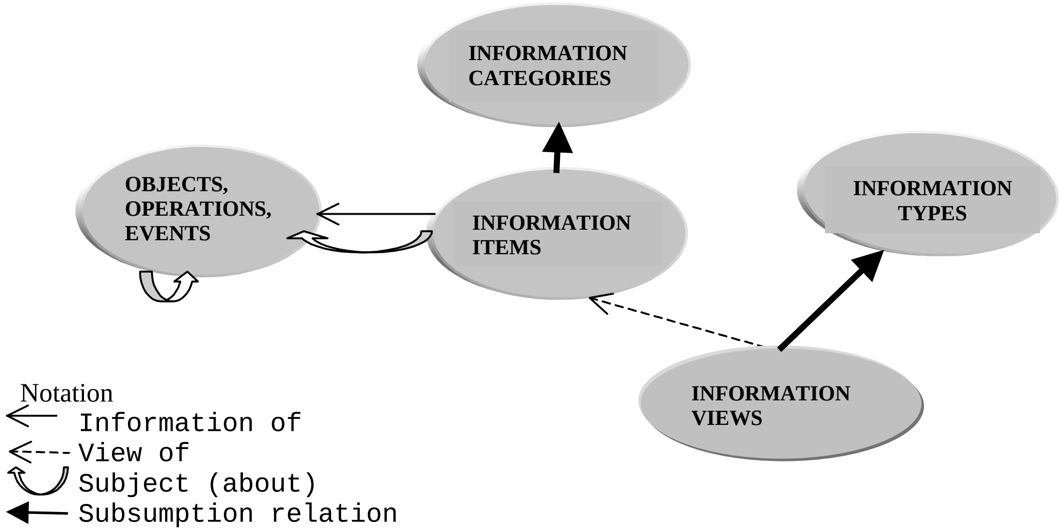


Figure 3: *General Structure of the Knowledge Base.*

Concepts, i.e. classes of objects and events, as well as concrete objects and events of the subject domain have a number of information items associated with them. For instance, the SET KEY in Figure 4 is associated with 4 information items: SET KEY LTERM, SET KEY GTERM, SET KEY POS, and SET KEY F2. Information items represent medium-independent pieces of information about domain aspects. They are categorised to information categories according to their role and scope in achieving communicative goals. For instance, as shown in Figure 4, SET KEY LTERM is a Linguistic Term (LTERM) with role:identifies and scope: naming. The SET KEY POS item is a POSITION INDICATOR with role:indicates and scope:position. Information categories, such as LTERM and POSITION INDICATOR, correspond to rhetorical acts [Mayb93][Andre93]. The intentions of presenting items that belong in these categories are identical to the intentions of the corresponding acts. Information categories suitable for technical documentation have been identified in [Vou99a]. These correspond to Rhetorical Structure Theory relations [Mann89][Andre93] and can be further extended with categories that are suitable for covering other documentation purposes. For instance, [Esp96] and [Car99] identified information categories for the documentation of object oriented software development methods and for medical guidelines, respectively.

An information item is associated with a number of alternative information views. Views realise information using a specific medium and format, and are classified under information types that represent generic forms of information. Information types, and consequently information views, combine the sense of realising an information item (label, list, plot, symbol, portrait, information) with the actual medium of information presentation (text, graphics, image, information, video, audio). Returning to our example, the SET KEY POS item has two views: SET KEY POS TIND and SET KEY POS IMIND. SET KEY POS TIND is a text fragment that provides information about SET KEY POS. It is an instance of the INFO TEXT TYPE, information type. The SET KEY POS IMIND is an instance of the PORTRAIT IMAGE TYPE and represents an image that is a “portrait” of the SET KEY POS item. Views may also include characteristics that express regularities in data sets and are necessary for selecting a medium/modality for realising information. These are:

dimension, transience, ordering, scalability and continuity [Arens93]. Furthermore, information views are associated with user-oriented information that constrains their selection during presentation. User-oriented information, as it is indicated in Figure 4, may contain information about language (for the lexicalisation of terms and textual views of items), user type indications, as well as hardware and software platform requirements for the corresponding views to be effective in an efficient presentation. User type indications may include user categories, user abilities, specific needs and interests. Users may be categorized based on their background knowledge (expert vs. novice) and on their generic needs (user vs. technician) on the subject domain.

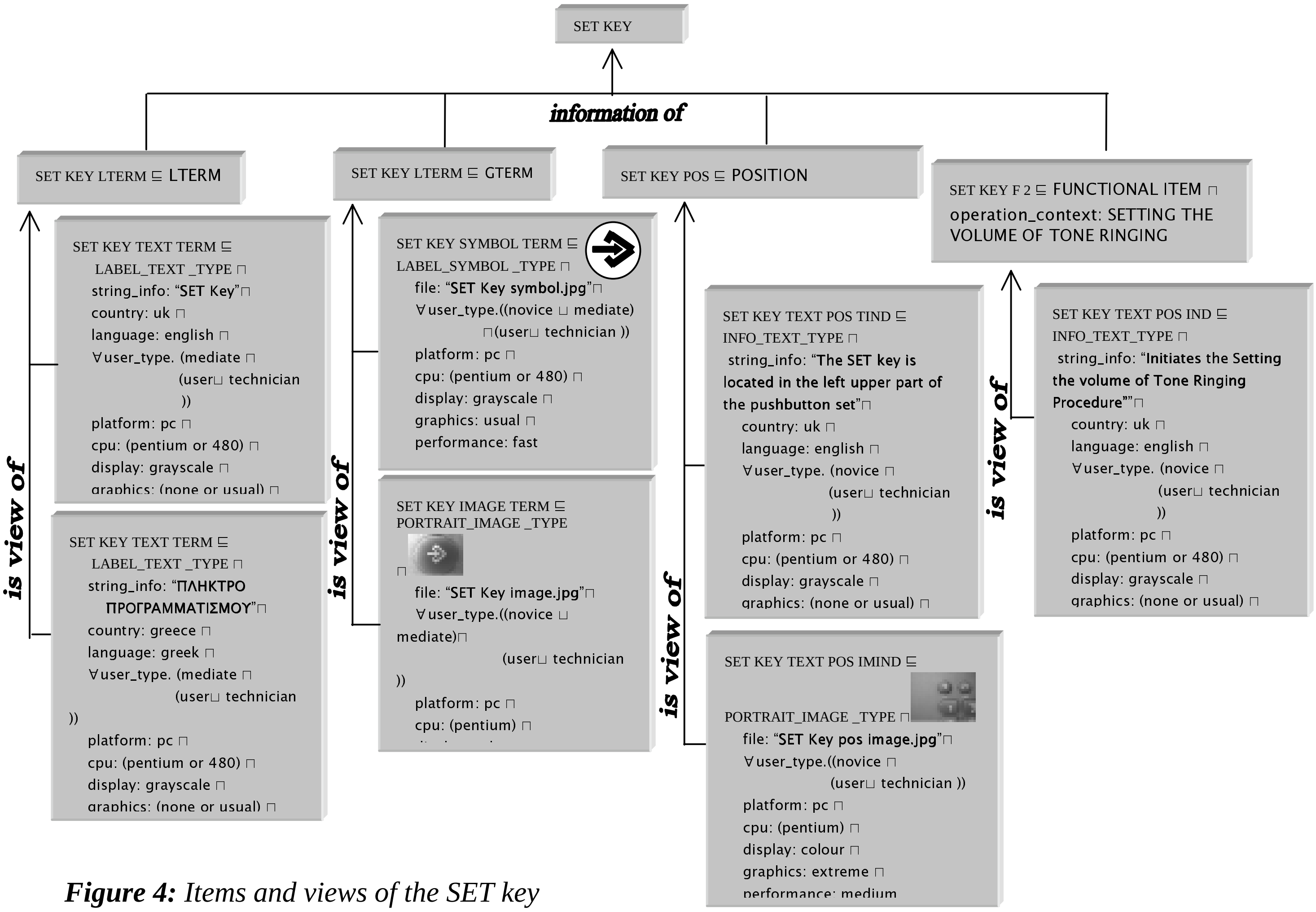


Figure 4: *Items and views of the SET key*

Domain concepts and objects, as well as information items, are classified by subject, i.e., by the aspects that they deal with. For instance, the SET KEY is classified in the following subject categories, among others: `SET KEY` (itself), `SETTING THE VOLUME OF TONE RINGING` (via the `operation_context` of the functional item `SET KEY F2), TONE RINGING and VOLUME` (which is the subject of the `SETTING THE VOLUME OF TONE RINGING).` Interest in these domain aspects may designate interest in information about the `SET KEY.`

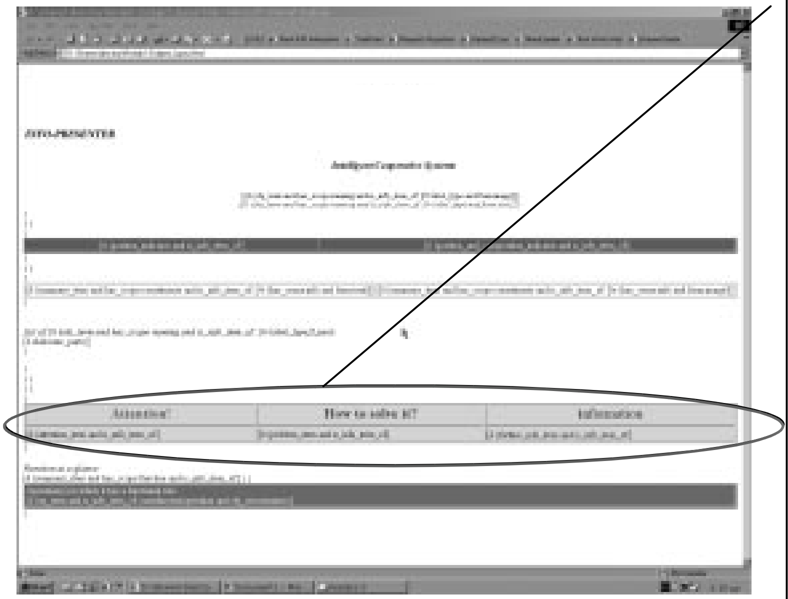
During presentation, a system must choose information items according to their scope and role in achieving communicative and presentation goals, and based on user characteristics and further contextual constraints, it must choose views of these items. Selection of an information item depends on the availability of views for realising this item in a presentation context, and utilization of views depends on whether the corresponding items achieve the presentation intentions. Distinguishing between items and views, the framework offers more flexibility and enables coordination of the information content selection and media design mechanisms.

Concluding, the proposed framework (a) provides the basis for tailoring information presentations, (b) facilitates the specification of generic information categories and types of views for achieving presentation intentions and realizing information in various presentation contexts, respectively, and (c) provides the building blocks for information producers to specify both, the information items related to domain aspects and alternative views for realizing these items in various contexts of information use.

In INFO-PRESENTER, the presentation task is driven by layout specifications.

Layout Database

Layouts are encoded as extended HTML pages with optional sections of information.



```
{<table BORDER COLS=3 WIDTH="100%" BGCOLOR="#F0E233" >
<tr>
{
<td><center>
<b><fontcolor="#FF0000"><fontsize=+3>Attention!</font></font></b><
/center>
<hr WIDTH="100%"><font color="#000000"><font size=+1>
        [/i (attention_item and is_info_item_of:]
</font></font></td>
}
{
<td><center>
<b><font color="#990000"><font size=+3>How to solve
it?</font></font></b>
<hr WIDTH="100%"></center>
<font color="#000000"><font size=+1>
        [/i (problem_item and is_info_item_of:]
</font></font></td>
}
{
<td><center>
<fontsize=+3><b><fontcolor="#990000">i</font></b>
<fontcolor="#660000">nformation</font></font>
<hr WIDTH="100%"></center>
<font color="#000000"><font size=+1>
        [/i (further_info_item and is_info_item_of:]
</font></font></td>
}
```

***Figure** 5: Layout Specification for Objects' Presentation*

An example of such an HTML page is depicted in Figure 5. Optional sections are included in brackets ({}) and information item sections are included within square brackets starting with "/i". Information sections may also include view sections that drive the system to select views of a specific information type. Information view sections are included within square brackets starting with "/v". A formal BNF specification of optional information sections is as follows:

```
<information_section>::<information item>|<information view>|<optional item>
<information_sections>::<information_section>|<information_section><information_sections>
<information item> :: [/i <term><information view>]
<information view> :: [/v <term>]
<optional item> :: {<string><information_sections><string>}
```

A `term` may be a description of an information item category, of an information view type, of a class of information items, of a class of information views, as well as the description of a specific item or view. For instance, the following term represents a textual view of an information item that summarizes the constituents of the object that is in the focus of attention.

```
[/i (summary_item and has_scope:constituents and is_info_item_of: focus_of_attention
```

```
    [/v (has_sense:info and form:text)]
]
```

Furthermore, `terms` may be macros that elaborate on specific items and views. An example of such a macro is the `elaborate_parts` shown in Figure 5, that returns a list of views of the constituents of an object, or of the steps of an operation. Macros help specifying complex interactions of information items and views, combined with special presentation requirements.

Optional sections allow adaptation of presentation and, more than information item/view sections, support the specification of alternative presentations. For example, let us consider the following optional part:

```
{<B>Information about A:</B>
   {<BR>Please read:<I>[/i A [/v form:text]]</I><BR>}
   {Hear that! - [/i A [/v form:audio]]}
<B>Bye!</B>},
```

where, A is a term.

In case A has a textual view that satisfies user characteristics, and does not have an auditory one, then the presentation will be as follows:

Information about A:
Please read: <A's textual description>
Bye!

We must notice that in case there is no view that can substitute any information item/view section of an optional section, then the optional section does not appear. This is the case with the auditory part of the presentation in the previous presentation. However, when there is a view of at least one information section, then the whole optional part appears. This is for instance the case with the whole optional section of our example.

Continuing our example, in case that A has an auditory view but does not have a textual one, then the presentation will be

Information about A: Hear that! – **Bye!**

and information about A will be heart.

However, in case that no view of A fulfils the required characteristics, then none of the included optional parts will be presented. Consequently, the inclusive optional part will not be presented as well.

It should be noticed that information view sections might indicate the intended audience of an information item, by including a specification of users characteristics. In this way, information producers can further constrain the contexts in which specific information items are presented, or the contexts in which specific views are used. Therefore, combining user-oriented information specified in information views, with information about the intended information consumers specified in information view sections of layout specifications, it is conjectured that information producers can control the amount and form of information in various contexts, and information consumers are given a great flexibility in satisfying their idiosyncrasies.

Layout specifications may be considered to realize presentation plans or schemes. Information sections correspond to sub-plans and information item and view sections to rhetorical and realization acts, respectively. Since information items correspond to rhetorical acts, the intention of an item is identical to the intention of its corresponding act. In this way, we can recover the intentional structure of a layout specification. But, this is a subject of further study and research.

Layout specifications are related with conditions. When these conditions hold, then the system selects the layout to guide subsequent information presentation. Therefore, the generic structure of a layout specification is:

`L=<C,H>`

Where, C is a set of conditions that must be satisfied in order the layout to be selected for presentation, and H the extended HTML part that drive information presentation.

Currently, INFO-PRESENTER exploits layouts for objects' and for operations' presentation. However, given that each of these layouts have 12 optional parts, the total number of possible presentation alternatives for each layout, are

$$\sum_{i=1}^{i=12} C(12,i) = 4095$$

This number must be multiplied by the number of alterative views of each information item.

This type of layout specifications is in close spirit with specification approaches in AVANTI [Fin99], AHA [Bra98], "formation" [FORM] and are within the scope of the ICONOCLAST [ICON] project. However, in contrast to most of these approaches, the information producer does not have to specify exhaustive conditions in optional parts and all the conditions of all possible alternatives, in each part of the presentation. On the contrary, the user may choose the appropriate level of abstraction to make the specification. This may range from very abstract information categories (leaving the type of information unspecified), to specific views of specific items with detailed user-oriented data. Furthermore, to construct layout specifications, producers must consult the represented knowledge, the characteristics of potential users specified in the user model, and must be driven by the ontological commitments (information categories, information types and their relations) of the formal representation framework described above.

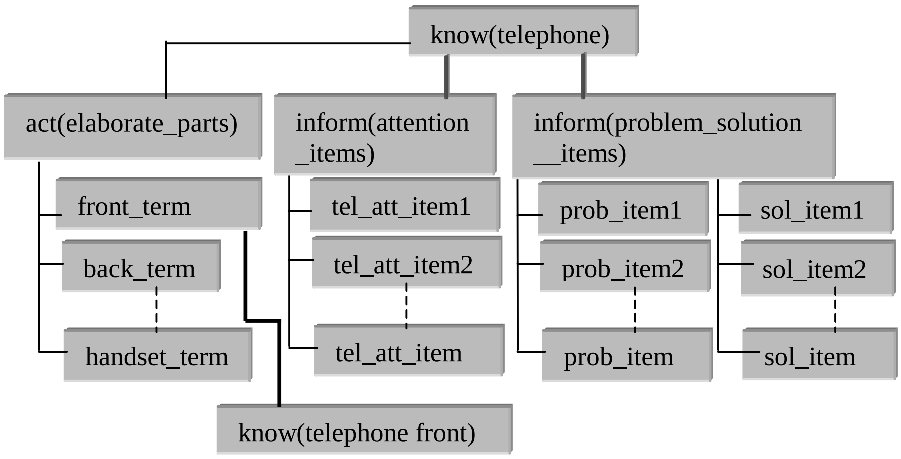


Figure 6: *The structure of the Discourse Context.*

Discourse Context.

The Discourse Context provides a blackboard facility in which the actions of the system and the user are recorded. As shown in Figure 6, the dialogue context has the form of a recipe graph [Loch96]. The system inserts the utilized recipes, i.e. the presentation strategies realized by the layout specifications. For instance, in Figure 6, the actions (`act(elaborate_parts)`, `inform(attention_items)`, `inform(problem_solution_items)`) realize a strategy for providing information about the object `telephone`. These actions are further refined by the selection of information items and information views for the actual presentation. When the user clicks on a telephone part (e.g. telephone front) then, this action is recorded in the discourse context (indicated with a bold line). In this way, the dialogue context is gradually constructed.

The dialogue context enables the system to locate the focus of attention, to infer the information needs, interests, and knowledge acquired by the user, and to participate in consistent and coherent dialogues with users. In the current implementation of INFO-PRESENTER, the dialogue context is utilized for locating the focus of attention and for presenting information consistently.

User Model

Currently this is a set of facts concerning user abilities, background knowledge and preferences about media-modalities. It is associated with simple update and maintenance procedures.

The user model is updated from the answers of users to a questionnaire displayed in the start of a dialogue session.

Concerning knowledge about domain aspects, it is assumed that when information items are displayed, then the user acquires the presented information. The user model is updated with new facts indicating the information acquired by the user.

INFO-PRESENTER modules are as follows:

Goal Formulation

The task of this module is to convert messages into goals that are understandable by the system. There are two types of messages: External messages from the user (when the user requests information about an aspect), and internal ones from the layout design and goal refinement modules.

Goal Refinement

This module refines the goal statement forwarded from the goal formulation module with user characteristics. It further recognizes whether the goal is about an object or an operation, and whether it concerns (a) the aspect itself, (b) a macro, (c) an information category concerning that aspect, or (d) an information view type of an information category.

In case (a) the system directs the request to the layout design module.

In case (b) it decomposes the macro to a set of goals of type (b), (c) or (d) and directs them to the goal formulation module.

In case (c), the system directs the goal to the content selection module and finally,

In case (d), the system directs the goal to the media design module.

Layout Design

The layout design module determines the layout specification that should be used for presenting the requested information. Towards this goal it checks the conditions of layout specifications. In case conditions are satisfied, a layout is selected and the module parses the extended HTML text. For each information section, it issues a request to the goal formulation module.

Based on the input from the media design module, the layout design determines whether optional information sections shall be presented and substitutes information sections with the determined information views. The output is forwarded to the layout realization module.

Content Selection

The content selection module retrieves and filters information items that belong to the requested information category of a domain aspect.

Questions about information items issued by the content selection module can be considered to correspond to rhetorical acts, given the correspondence between information categories with them.

The content selection module issues requests concerning information types of the selected items, to the media design module.

Media Design

The task of the media design component is to select the appropriate views of the information items mentioned in the request. The request may also state the particular information type to which these views must belong.

In any case, the module checks whether each selected view satisfies the user characteristics (abilities, preferences, background knowledge), whether it is consistent with previous presentations of the same item, with presentations of related items and determines the number of views that shall be presented for each item.

Selected views are forwarded to the media realization module and to the layout design module.

Media Realization

Based on the information views selected by the media design component, the media realization module retrieves the selected media objects and adds stylistic guidelines (HTML tags) for their realization. For instance, a list of views may be itemized or be tabulated, certain text segments may appear in bold or italics, with a particular colour, or an item may be formed as an anchor.

Layout Realization

The task of this module is to merge the results of the layout design and media realization module. The result is an HTML page that is sent to the client.

The generated document is presented using standard WWW software (e.g. a standard web browser).

4. Implementation and current status

INFO-PRESENTER has been implemented using SWI and AMZI! Prolog. The system uses the AMZI! Logic Server for interaction with CGI and the WWW Server. The representation framework and system's knowledge base have been developed using the BACK Description Logic. Currently, the knowledge base contains concepts, information items and views describing parts of a telephone device, as well as tasks operating such a device. Figures 7-11 show presentations generated by the system during a dialogue session. All presentations contain information for a novice that prefers text to other forms of information.

The presentation of the telephone device is given in Figure 7. We must notice the correspondence of the presentation with objects' presentation layout specification. The system presents the label items for the telephone device, summary items about its constituents, a list of all telephone parts, attention items, problem items and solution items, as well operations in which this object participates.

Continuing the dialogue with the system, the user requests information about the telephone front. Information about it appears in Figure 8. This includes just labels and telephone front parts.

Presentation of the speaker, as Figure 9 shows, includes only a text label and a textual description of speaker's position.

For the presentation of information about operations, the system exploits the corresponding layout specification. Examples are shown in Figures 10 and 11.

Currently, INFO-PRESENTER supports mostly adaptation of content. Adaptive navigation support is currently supported in a restricted way: As already mentioned, in case the system has presented a view of an information item, it assumes that the user has acquired the presented information. In subsequent presentations, the same information is replaced with a link to the information. Links to this information, in subsequent presentations, are hidden. However, we already study and implement more robust mechanisms for link annotation and hiding, exploiting user interaction with the system. We plan to integrate this feature in INFO-PRESENTER within the next months.

5. **Concluding Remarks**

The paper presented on-going research towards generic tools and methods for fulfilling the combined needs of information users. It presents the principles, framework and key issues of our research, and focuses on developments concerning INFO-PRESENTER. This is an interactive system that provides knowledge-based and layout-driven information presentations, intending to satisfy the idiosyncrasies of information consumers, supporting a great amount of tailoring information to their needs, interests, preferences and background knowledge.

The aim is to provide a comprehensive knowledge representation framework that will guide information specification efficiently, and upon which, efficient and effective information adaptation will be based. The proposed framework is general, extensible and makes the ontological commitments, which are crucial for supporting both of these tasks.

We already designed and start implementing an intelligent authoring tool that will help information producers to make information specifications and produce the required knowledge bases efficiently. However, we must point that even without this tool, information specification is a straightforward task, but of course not for, let us say, technical writers.

Layout specifications have been kept as simple as possible, in order to facilitate their efficient production and their effective utilization during information presentation. However, information producers will need tools for the production of these specifications. This, in accordance with support provided for combining layout specifications in a way similar to the combination of plan operators, is within our future plans.

It is our belief that although this set of tools can empower both producers and consumers of information, their scope is limited. This mostly concerns INFO-PRESENTER: It may support interaction in a high degree and demonstrate the value of the methods and techniques proposed, but it will be difficult to support cooperative behaviour and provide helpful behaviour. To do this, the system must be able to reason about communicative goals, and presentation intentions, taking into account users' mental state. Furthermore, it must enable cooperation between media components, between media and content selection components, reason about user intentions and goals, and tailor information to these settings effectively. Towards this aim we conduct research in agent-based cooperative systems [Vou99b].

References

[Arens93] Y.Arens, E.Hovy, M.Vossers, “On the knowledge underlying multimedia presentations”, in Intelligent Multy Media Interfaces, M.Maybury (Ed.), 1993.

[Mayb93] M.T.Maybury, “Planning Multimedia Explanations Using Communicative Acts”, in Intelligent Multy Media Interfaces, M.Maybury (Ed.), 1993.

[Andre93] E.Andre, R.Rist, “The Design of Illustrated Documents as a Planning Task”, in Intelligent Multy Media Interfaces, M.Maybury (Ed.), 1993.

[Spe99] M. Specht, ACE - Adaptive Courseware Environment, http:/129.26.167.17:8080/ACES/html/NRHM/NRHTML.html

[Bru98] P.Brusilovsky, “Adaptive Educational Systems on the World Wide Web: A Review of Available Technologies”, in Proc. of workshop “WWW-Based Tutoring” at 4th Intern. Conference on Intelligent Tutoring Systems (ITS’98), 1998.

[Vou99a] G.Vouros, “Conceptual Modeling of Multimedia Objects for Intelligent Information Presentation”, submitted for publication.

[Vou99b] G.Vouros, “Towards a Generic Architecture for Cooperative Learning Environments”, International Journal of Continuing Engineering Education and Life0Long Learning, to appear.

[Loch94] K.E.Lochbaum “Using Collaborative Plans to model the Intentional Structure of Discourse”, Center for Research in Computer Technology, Division of Applied Sciences, Harvard University, TR-25-94.

[SRM98] M.Bordegoni et. al., “A Standard Reference Model for Intelligent Multimedia Presentation System”, Computer Standards and Interfaces, 1998.

[Rei94] E.Reiter, C.Mellish, J.Levine, “Automatic Generation of Technical Documentation” Applied Artificial Intelligence, volume 9, 1995.

[Esp96] F.Espinoza, K.Hook, “A WWW Interface to an Adaptive Hypermedia System”, PAAM, April 1996, London.

[Car99] B. de Carolis, F. de Rossis, “The Dynamic Generation of Hypertext Presentations of Medical Guidelines”, The New Review of Hypermedia and Multimedia, Vol 4, 1999.

[Bra98] P. De Bra, L.Calvi, “AHA: a Generic Adaptive Hypermedia System”, 2nd Workshop on Adaptive Hypertext and hypermedia, HYPERTEXT’98, Pittsburgh, USA, June 20-24, 1998.

[Fin99] J.Fink, A. Kobsa, A.Nill, “Adaptable and Adaptive Information Provision for All Users, Including Disabled and Elderly People”, The New Review of Hypermedia and Multimedia, Vol 4, 1999.

[FORM] Formation, http://www.aiai.ed.ac.uk/project/formation/formation.html

[Par96] C.Paris, K.Vander Linden, “An Interactive Support Tool for Writing Multilingual Manuals”, IEEE Computer, July 1996.

[ICON] ICONOCLAST, http://www.itri.brighton.ac.uk/projects/iconoclast

[Mann89] W.C.Mann, C.Matthiessen, S.Thompson, “Rhetorical Structure Theory and Text Analysis”, ISI Research Report, ISI/RR-89,242, 1989.

[Zhou98] M.X.Zhou, S.K.Feiner, “Automated Visual Presentation: From Heterogeneous Information to Coherent Visual Discourse”, Journal of Intelligent Information Systems, Kluwer 1998.

[Wel96] C.A.Welty, “Intelligent Assistance for navigating the Web”, Proc. Of the 1996 Florida AI Research Symposium (Flairs 1996).

[BACK] T.Hoppe et al., “BACK Tutorial and Manual” Technische Universitat Berlin, Project KIT-BACK, March 1993.

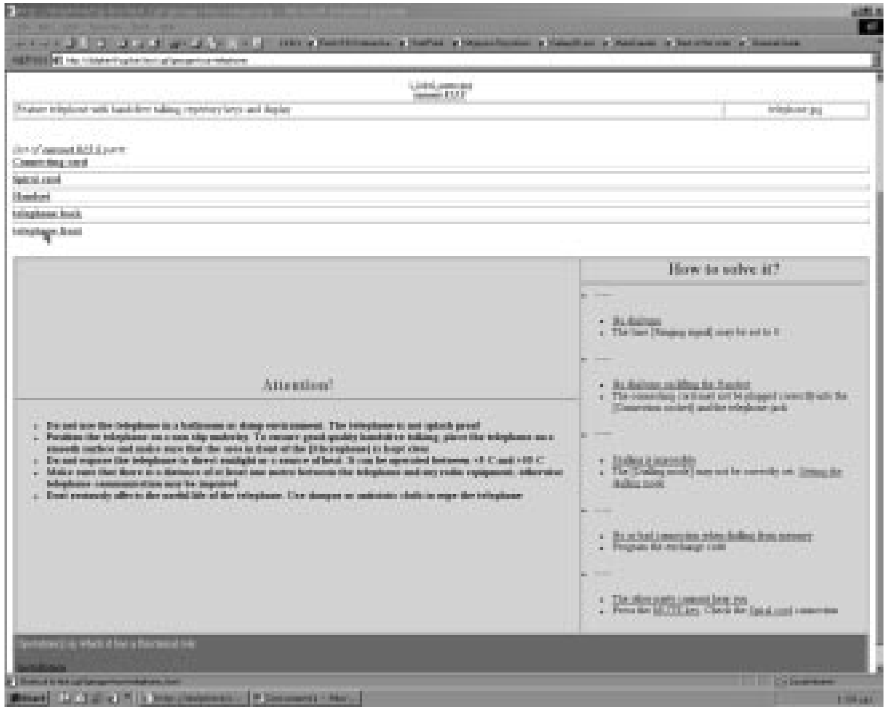


Figure 7

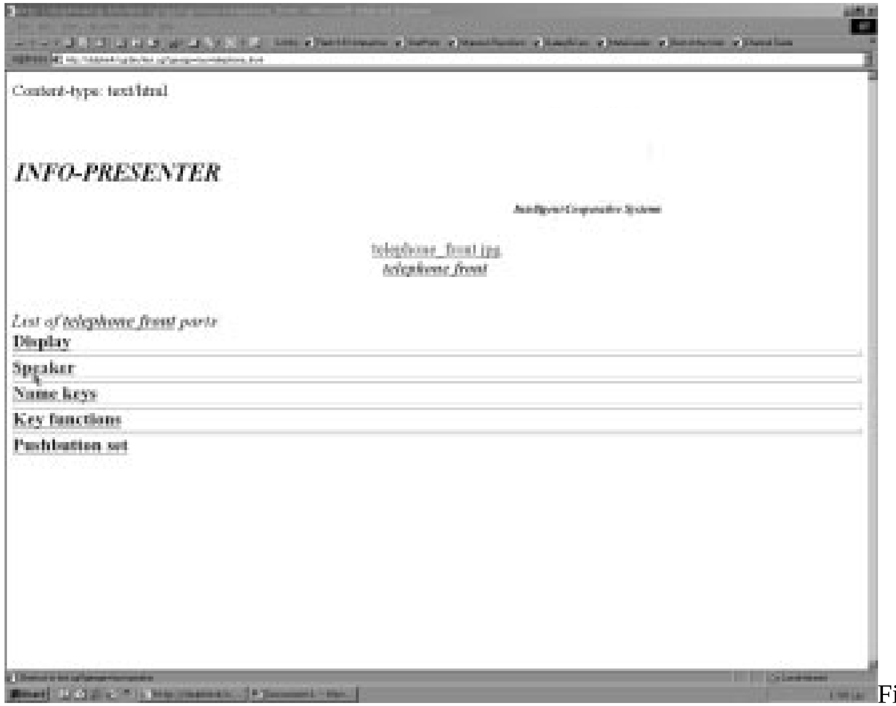


Figure 8

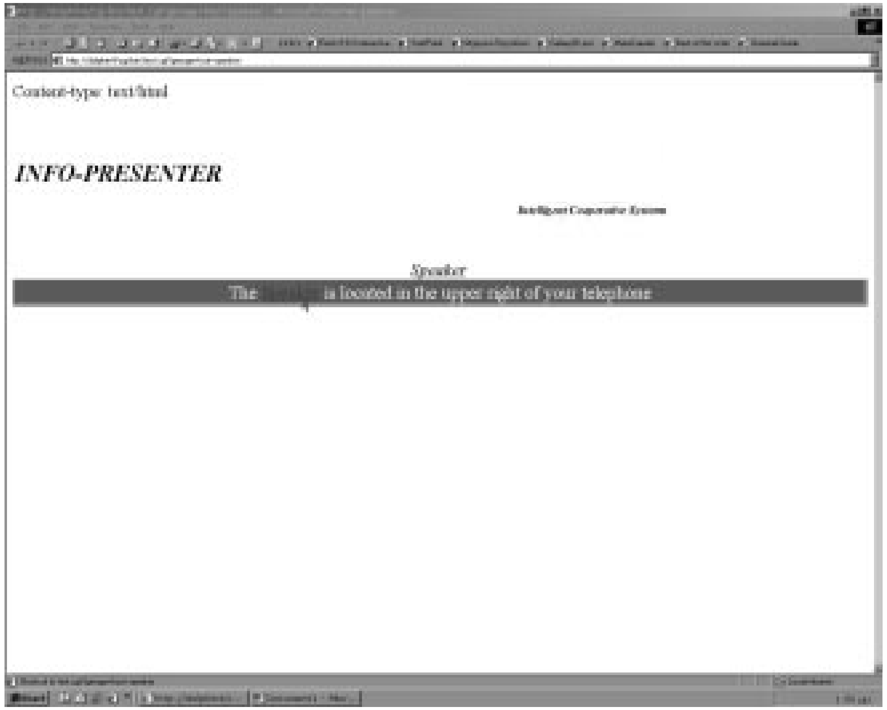


Figure 9

Figure 10


INFO-PRESENTER
Intelligent Cooperative Systems
information

Figure 11

Learning from the Art: The Art on the User Interface

M. Gea
Dpt. Lenguajes y Sistemas Informáticos
ETSI Informática. Universidad Granada.
mgea@ugr.es

J. Rodeiro
Dpt. Lenguajes y Sistemas Informáticos
ETSI Informática. Universidad Vigo.
Spain
jrodeiro@uvigo.es

Abstract

The computer is a tool used to accomplish goals. The computer programs are creative task, and the user interface is the way to achieve to the goals that the computer programs perform. In this context, the user interface should play a creative role too, mixed up with the proper work itself. This paper proposes an alternative way of building user interfaces according to this concept, and applying techniques widely used in art.

Keywords

Visual Design, Interface Metaphors, Adaptable User Interaction

1. Introduction: The role of the User Interface

Computer is a common tool on the human everyday's life, and is used to achieve goals when performing computer-tasks. The User Interface (UI) is the layer communication between the user and the computer, and therefore, it should adequate the computer features to the user's needs. This is not always obtained with usual applications because the time and costs to obtain this objective are not proportional to the required effort.

The UI is a living and dynamic layer which is intimately related to humans. The purpose of the UI is to isolate the user from the internal details of the computer systems, and to give assistance to the user in order to perform their tasks easily. Therefore, the UI becomes closer to the user and it should be comprehensive to the human's mind. For such reasons, we have to design the UI independently from the computer programs, allowing the user to personalise the interface to his own way of thinking, working and perceiving the reality.

2. The creative process

The visual media is the vehicle to transmit ideas and concepts, the human reasoning, their creativity as well as their feelings[3].The creative activity involves a methodological process orientated to achieve a goal. To perform an UI we may use that methodology combined with visual variables. The steps followed in a creative process are briefly commented below.

- *The ideation phase* is the preliminary phase for any creative process. This initial step focuses on the assessment of a creative basis. For that objective, the artist uses different sources of inspiration which may contribute to the result, such as scientific sources (physic, biology nature, etc.) or human aspect (reasoning, perception).
- *The conceptual phase* is related to the identification of the ideas regarding to the graphic compounds, and the way in which they are collected during the methodological process. These ideas have to be expressed in some way over a

picture, sculpture of any expressive surface. The methodological process involves some restrictions/possibilities of expression, to open/close new formalities and to give methods to perform this. Therefore, we have different ways of expressing the same idea.

- *Creation of forms.* This step focuses on representing the object which constitutes the work by using modelling techniques. This creative phase involves the construction of the main graphic compounds of the resulting image. Every one forms part of the communication process, and the whole combination represents a homogeneous unity.
- *Analysis* is the study of the objectives of each one of them components and its support to the final work (from individual components to the global composition). This may be considered as a late stage in which the author performs a judgement on the final work and its adequacy to the initial purposes.

3. The User Interface Layer

Architectonic approaches have been proposed in the UI definition using the layer approach [2][4][1], but in many cases these approaches are dependent on the system and the use of toolkits to build the UI. Therefore, the resulting UIs have a similar apperance because they use the same standard components. We propose a layer approach that separates the UI from the computer program process. In this context, we define layer taxonomy according to the functionality of each layer.

- **Visual layer.** This layer defines the external appearance of the User Interface. It may be in principle static, but it can be modified with filters.
- **Content layer.** This layer defines the area in which the information is contained (input/output information). The limits of the display are a restriction for the availability of such space, but we can use other techniques which could help us to show relevant information. In any case, the shape, size or location can be restricted. Thus, these variables may be dependent on the context in which the information is showed.
- **Filter layers.** Filters are a special type of layer that modifies part of the interface. It may be used as a *mask*, preserving parts of the interaction activity or merely as a *filter*, modifying some aspects of the visual appearance (noise, contrast, enhancement, etc.). Filter can be applied to other visual layers to show temporal changes, according to a particular event in the system.
- **Behavioural layers.** This type of layer defines a semantic action of particular regions in the User Interface. These areas can be described as sensitive points with interaction abilities. The link defines a relationship between a particular event (focus in, movement, mouse click, etc.) and a related effect on the User Interface (change the appearance of the UI) or actions of the Core System (save, print, copy, etc.).

4. Conclusions

In this paper we have presented a new conception of the User Interface based on layers. This approach is rooted by the creative process used in designing and fine arts. By doing so, the User Interface can be improved, allowing to adequate different users and situations. Additionally, the proper creation of the User Interface has an artistic nature.

An author when designing a UI follows a similar methodology as for painting a picture. This may be useful to register the way of performming the task on the computer.

References

1. Arch, *"A Metamodel for the Runtime Architecture of An Interactive System"*, The UIMS Developers Workshop, SIGCHI Bulletin, 24(1), ACM, January, 1992.
2. Myers, Brad A. UIMSs, Toolkits, Interface Builders. Revised from Myers, Brad A. "User Interface Sofware Tools", *ACM Transactions on Computer-Human Interaction*. Vol. 2, nº. 1, March, 1995. Pp 64-103.
3. Limochi M. *"Variables creativas infográficas en el desarrollo de la imagen"*. Phd Thesis. Universidad de Granada.
4. Paff G.,Ten Hagen P.J.: *"Seeheim Workshop on User Interface Management Systems"*. Springer Verlag, 1985.

The Productive Information Society: A Basis for Sustainability

Wolf Göhring (wolf.goehring@gmd.de), GMD
Institute for Autonomous Intelligent Systems (AiS)

The 21C Problem of the Global Society

The central problem of the 21st century lies in the growing consumption of fossil fuel, of energy, of raw material for industrial products, and of agricultural areas for towns and industries, in the devastation of immense areas, in the greenhouse effect resulting from the emission of CO_2 and other gases, in the elimination of traditional social relations, and in the mechanical and informational rationalization of labor. Mastering this 21C problem men will alter considerably technical production and social relations which are founded in the way how men determine what is needed and how they acquire the needed things.

Computer supported collaborative work
helps to solve the 21C problem.
But how?

The Intertwining of Information and Labor

Labor is reflected socially by the work steps that fit together, technically by the parts forming an object and informationally by the technically supported descriptions of things and the accompanying communication. These three sides of labor are closely interrelated.

Negotiating and processing a thing or a system people use their knowledge, they also use written, drawn or electronically stored information. They talk about intended and produced changes which again is remembered in mind or written and drawn on paper or stored electronically. The social actors reflect their world in information through collective labor in order to react collectively on the world of their objects and on nature.

Information is reflected in labour and products and
labor is reflected in products and information.

The Web as the Central Tool for Free Organized Labor

With web support people of different skills and from different regions become able to collaborate closer on the basis of technically networked information. Breaks and gaps between worksteps may be overcome thus facilitating cooperation. People will overcome the traditional division of labor, especially the division into white and blue collar work and the taylorism. They don't need to act under predefined orders.

Though ever more complex the technical world is reflected in the web at its best. The *universal access* to appropriate data and methods as well to data and documentation of various systems makes it possible to take care of resources and to elaborate innovative things efficiently. Thus the web mediates in a new way the acquisition of existing or emerging technologies and the generation of new ones giving a basis for a sustainable evolution around the networked world. The web augments the means in communicating data and behavior of the underlying objects. It becomes the externalized knowledge of mankind retrievable by and communicable to everybody.

The web gives the technical basis for a coming productive information society.

Alienation of Work Vanishes in the Productive Information Society

With web support and in relation to accessible machinery, resources and needs individuals determine collaboratively the concrete labor each of them will do. In this negotiation process the *universality of each individual* will consist in the ability and in the *freedom* to retrieve suitable information from really every point of the global society and to organize collaboratively his own productive labor and to enrich the web with information. People define their *concrete equivalent* product, service and contribution such that after production there is no need to exchange the product on the basis of its value and to determine this value *ex post*. Aristotle's principle: „In order to maintain society nature decided that there are leaders and retinue. He who is able to foresee wisely, whom made nature to be leader and ruler, he who is able to execute commands with his hands' work, whom decided nature to obey and to be servant" is changed to „The laws of his own social action, hitherto standing face to face with man as laws of nature foreign to, and dominating him, will then be used with full understanding, and so mastered by him. Man's own social organisation, hitherto confronting him as a necessity imposed by nature and history, now becomes the result of his own free action." (Frederick Engels, Anti-Dühring)

With web support men organize a global „builder's yard" where the psychological, economical, and legal alienantion of work will vanish.

Issues

On acting as sketched out the 21C problem will be treated succesfully. But if we want to do so we find great lacks in computer science, in social relations, in economy and in legacy.

Issues in Computer Applications

In the same period in which computer power is doubling industrial productivity grows far less than 10%. Today's instruments are not adequate to operate efficiently in the growing jungle of information, to handle the permanent collisions between plan and reality, to marry and to divorce people, things and worlds causing conflict and confusion. These issues meet with usability, with ergonomic, psycological and physiological requirements and construction principles of computer systems.

Social Issues

Formally, the social actors in the information society are not restricted to an elite. But practically, productive labor has to be adopted in the ideas of the information society. Workmen and, above all, blue collar workers that behave on their own have to be admitted in the models, not only being an object of computer supported planning. Education, training and efficient computer tools that are obviously suited for an emancipative usage on the job have to encourage blue collar workers to engage in this development. Schools, universities *and* practical work have to be connected through the web. We need concepts for teaching the productive information society during the education of young people in schools and in enterprises, the training of craftsmen and workmen, and the retraining of unemployed persons.

Economical Issues

Solving the 21C problem the use value of products has to be considered globally, in long terms and in relation to nature *before* an object is produced. The web enables this doing, but competitors have to retain information in order to be winner on the market. They have to put the exchange value in the foreground instead of a fully determined use value. Competitors appropiate the product of their workers and sell it being the only way to realize and to evaluate the invested capital. But this proceeding excludes the free discussion on the use value and on the direction of the production which, supported by the web, consumers and producers could organize by themselves. We need public funding to organize steps into the productive information society.

Legal Issues

If people want to act freely in web and production oriented structures they observe a lot of formal bounds. Usual contracts of employment give them no equal rights in accessing the information and the productive inventory in society.We need a political and legal framework such that people don't lose the equality of their rights during their life, a framework that supports and legalizes the descripted free use of information, that adopts and enhances the outlined possibilities of the web, and that stabilizes social security in the productive information society.

Conclusion

The solution of the 21C problem strongly intertwines with a
free extebding and efficient use of the web by all men.
free, extending and efficient use of the web by all men.

Dialogue Interface for Programming in Prolog

Adriana Jergová
Faculty of Informatics, Masaryk University,
Botanická 68a, 602 00 Brno, Czech Republic
e-mail: ada@fi.muni.cz

Motivation

In the last few years spoken interfaces are often used in the telephone-based querying systems, voice-controlled systems, etc. For the visually impaired users or users with other disabilities they can bring new opportunities to overcome some of the difficulties resulted from their handicap. At Laboratory of Acoustics and Phonetics at Masaryk University we are developing programming environments for various languages that facilitate the task of programming for blind programmers as the conventional environments are extremely inconvenient for them [4, 5].

Here we present a conception of spoken programming environment for Prolog language that is designed for visually impaired programmers. Employment of spoken interface in the applications where graphical interface is serviceable needs special techniques and strategies to be adopted. In the case of programming by means of spoken interface the communication between the user and the system involves several interfering areas and levels—input of text (source code), speech informational feedback providing the source code overview, system commands, help (explication, guidance), and metacommunication (clarifications, repairs).

In this paper we concentrate our attention to the first two mentioned areas and we suggest the representation of the treated text (programs in Prolog) which allow us to exploit advantages of the spoken input and to compensate impossibility of visual checking. The main goals are

- elimination of syntactic errors,
- reduction of semantic errors and support for their search and correction, and
- support for good orientation in the program source.

Sequential Syntactic Input

A straightforward way of entering program source could be providing a user with possibility to dictate the text in sequential manner "word by word" or "character by character" imitating thus classical keyboard input. This approach without additional analysis does not support good orientation in text and programs created in this way would be very error prone. On the other side, this method can be applicable in some cases, for example when the text is completely prepared and does not require modification or consists of only short sequences without nesting, sequences with atypical or without any structure. This mode of input may be employed also for unexperienced single-session users, etc.

Structured Syntactic Input

More natural and safer way would be entering of semantic units of reasonable granularity having precise syntactic form constructed automatically. To determine the size and the structure of input units as well as their hierarchy we must take into account

- grammatical structure of program sentences – in order to preserve their original structure and to provide correct syntax of the final output text
- semantic sectioning of the program – in order to allow convenient organization of the program and to facilitate orientation in the entered text for the user
- human capabilities – in order to minimize memory load of the user.

Semantic Input

Although compound term syntax is the same (with exceptions like infix form for operators, list notation, etc.) on its various positions in the language constructs, treating terms differentially (i.e. using term's context information in naming of processed particles) supports better orientation in the text. Another helpful strategy is the employment of meaning of the term elements. This can be realized implicitly via intelligent variable and atom naming. This is to be done by the user, but the system can support and exploit proper naming and thus facilitate orientation in term during its creation, reading, or modification as well as searching and checking of interpredicate relations, reducing thus the risk of semantic errors. Meaning can be introduced also explicitly by using *field names* for the term arguments in the style of ECL^{i}PS^e [2] library `structure.pl` or by means of predefined sentence patterns.

At the present stage of the system development, the meaning of term arguments must be provided by the user itself and therefore it is always a trade-off in efficiency of input and efficiency of later processing. Providing additional information is recommended mainly for terms representing predicates or often used data. It is subject of the future research, how to detect arguments and approximate their meaning automatically from input sentences.

Conclusion

The third style of communication appeared to be the most effective in the series of WOZ-like trials[1] we have performed, but it has the highest demands on speech recognition. In the case of using noncontinuous speech recognition, the structured input will give the best results. In addition, this type of input can be naturally converted to In our tests the programmer was not allowed to look at the display and all the phases of the programming process were realized by conversation with a computer operator. Programmers have known that the operator is human and understands Prolog programs. multimodal input. The sequential input should be supported also as an option, as it can be useful in special cases.

Acknowledgement

The research has been partially supported by Czech Ministry of Education under the Grant VS97028 and by Grant Agency of the Czech Republic under the Grant 201/99/1248.

References

[1] Bernsen, N.O., Dybkjoer, H., and Dybkjoer, L.: What should your speech system say?; IEEE COMPUTER, 30(12) 1997, pp. 25-30
[2] ECL^{i}PS^e User Manual, Release 4.0, ECRC 1998
[3] Kopeèek, I.: Modeling of the Information Retrieval Dialogue Systems; Proceedings of the Workshop on Text, Speech and Dialogue - TSD'99, pp. 302-308
[4] Kopeèek, I.: Dialog Based Programming; Proceedings of the First Workshop on Text, Speech and Dialogue - TSD'98, 1998, pp. 407-412.
[5] Kopeèek, I., Jergová, A.: Programming and Visually Impaired People; in Proceedings of ICCHP'98, Wien-Budapest, September 1998, pp. 365-372.
[6] Mooney, R.: Learning for Semantic Interpretation: Scaling Up Without Dumbing Down, In Proceedings of LEARNING LANGUAGE IN LOGIC (LLL) WORKSHOP, 1999
[7] McRoy, S.W., Ali, S.S., Restificar, A., and Channarukul, S.: Building Intelligent Dialog Systems, to appear in Intelligence, 1(10), ACM Spring 1999
[8] Waern, A.: What is an Intelligent Interface?
http://www.sics.se/~annika/papers/intint.html

A Universally Accessible ATM or Information Kiosk

Ing-Marie Jonsson, Neil Scott
The Archimedes Project, Stanford University
210 Panama Street
Stanford, CA 94305, USA
E-mail: *ingmarie@csli.stanford.edu* *ngscott@csli.stanford.edu*

ABSTRACT: Based on the assumption that information is neither accessible or inaccessible - it is the form in which it is presented that makes it one way or the other, the main focus is to provide access to computer based systems for users based on their needs, preferences and abilities. We present a paradigm, which decouples applications from input and output devices. This decoupling is the basis for personalized user interfaces and interaction models using alternative input and output modalities. The paradigm also provides access to standard versions of applications instead of having to adapt a specific version to each access solution.
As an example of a practical application of personalized access, we describe an information kiosk that accommodates a wide variety of users without requiring changes to the hardware or software of the kiosk. This is achieved through the use of the Total Access System (TAS), from the Archimedes Project at Stanford University, that provides the framework for connecting and using alternative input and output strategies for interacting with the kiosk.
KEYWORDS: Personalized access, decoupled interaction model, enabling technology, device profile, information kiosk, multiple modalities, adaptation, Total Access System, universal access

Introduction

Many individuals are becoming severely disadvantaged through their inability to use computers to access information. The reasons are many and include factors such as lack of appropriate education or training, physical or cognitive disabilities, and fear of using computers. We base our approach for implementing an accessible information kiosk and ATM on the TAS [5], a strategy that provides access to any computer-based device. TAS separates the access issues into three components: An **accessor -** a personal user interface, a Total Access Port (**TAP**) - a connection between an accessor and a computer-based device, and a **communications protocol** between the accessor and the TAP.

An Accessible Information Kiosk

Information kiosks and ATMs are now part of the fabric of society, unfortunately most disabled people are denied access these kiosks. For many years, banks have used the excuse that disabled clients have the option of using a live teller for their transactions. Now, however, ATMs and information kiosks are often installed in locations such as shopping malls where there are no live tellers. Also, many local, state and federal government agencies are making kiosks the primary source of information and anyone unable to access them is severely disadvantaged.

Current Solutions

Limited accommodations, such as instructions printed in Braille, are provided on some kiosks and ATMs. But only ten percent of blind people can read Braille, and even those who do know Braille still have no way to read the silent instructions on the ATM monitor. In some cases, ATMs are mounted low to the ground to meet the needs of physically disabled people in wheelchairs. Occasionally, ATMs or information kiosks present messages for deaf users as American Sign Language displayed in a small window on the monitor screen.

These examples only begin to demonstrate the diversity of individual needs and to show how assistance for one group may lead to conflicts for another group. For example, Braille labels are often unreadable on an ATM that has been lowered because blind users must be able to place their fingers upright to read a label. It is not uncommon to see a blind person kneeling before an ATM just to be able to use it. Another case is when a tall person tries to use a lowered ATM and cannot read the LED or LCD screen because of the angle.

Furthermore, the owners or operators of information kiosks or ATMs often use systems from a variety of manufacturers. This may lead to confusion for clients who encounter different systems with unfamiliar interfaces at different branches.

Our solution

Our design rule is that all access features that sensibly belong in the kiosk should be included as part of its basic design. All others should be external and the kiosk should provide access via a communications interface. Our proposed solution also offers two levels of accommodation: **Basic access** to unaltered information kiosks, **Enhanced and Personalized access** to modified information kiosks.

Basic access - unmodified kiosk: Based on the use of an accessor interacting with the kiosk through a TAP. Needs a TAP equipped with a communication port connected to the kiosk. Protocol A, between accessor and TAP is standardized as Alternative Interface Interaction Protocol, AIIP by NCITS [3]. Protocol B, between TAP and kiosk is proprietary and depends on manufacturer.

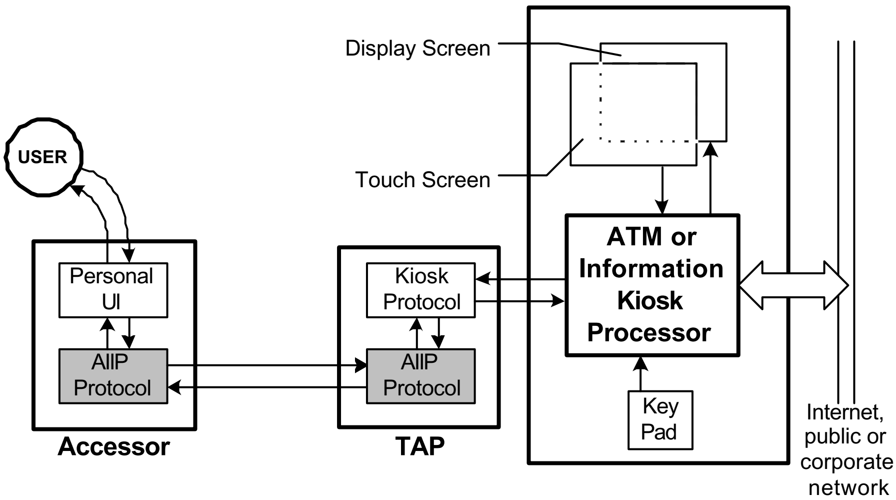


Figure 1. Basic Access

Enhanced and Personalized access - modified kiosk: All permanently attached peripherals on the kiosk, such as standard screens, keyboards and selector buttons, are configured as accessors to take advantage of our framework [2,3,5]. The separation of user interface from presentation and interaction model described in IAAP [2,3], enables ATM and kiosk owners to provide identical user interfaces on all of their machines, regardless of the age or version of individual systems or even if they are supplied from different manufacturers. The modified kiosk can accommodate the access needs of almost all users, for some users this means carrying a personal accessor, for others all capabilities are provided by the kiosk.

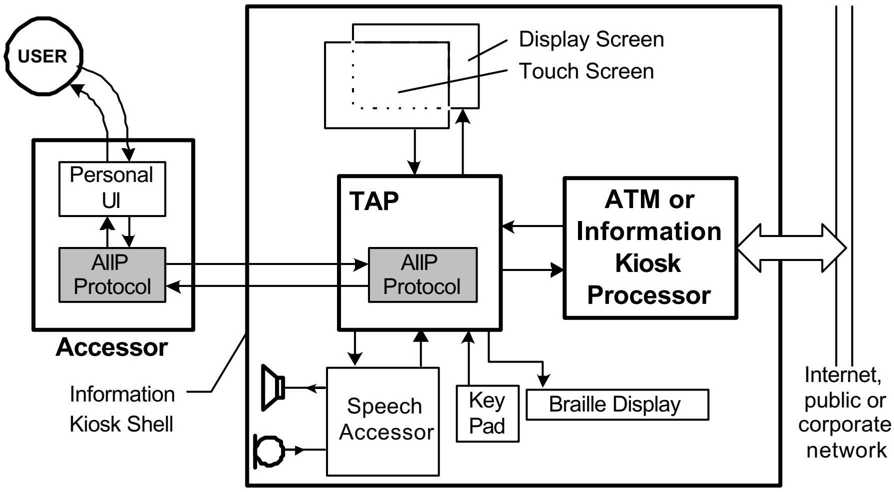


Figure 2. Enhanced and Personalized Access

Conclusions and Future Work

We have described a flexible system, where today's solutions will remain functional in the future, at the same time as it provides the framework for adding new interaction capabilities as the supporting technologies mature. The basic operation of this system decouples the user interactions from the information processing. This makes it possible to use alternative user input and user output strategies without adding to or changing the functionality or underlying hardware or software of the target system. Furthermore, the modified kiosk allows owners to provide a consistent user interface on all of their machines regardless of their source or age. This problem may be the reason for the meager accommodation options provided by current ATM and kiosk manufacturers [1,4]. Our solution thus also differs from approaches taken by other researchers who favor a purely integrated solution. [7]. It is interesting to note that another related research project KIMSAC, [6] which had a focus on a multimedia kiosk for social services, provided some accommodations based ergonomic design, handsets and speakers.

REFERENCES

1. Diebold Incorporated, (1999) Diebold Self Service Solutions, http://www2.diebold.com/products/selfserv.html
2. Jonsson, I. (1999), The Decopled Application Interaction Model, D.A.I.M,

Proceedings of HCII'99, Munchen.

3. National Committee for Information Technology Standards. Study group on Information Technology, Accomodation, (1999), http://w3.gsa.gov/web/m/cita.nsf/Portals/NCITS

4. NCR Corporation, (1999) NCR Self Service Solutions,. http://www3.ncr.com/product/financial/product/sss/ssindex.htm

5. Scott, N. (1991) The Universal Access System, American Voice Input/Output Society Conference, Atlanta.

6. Tierney, Mark (1998) KIMSAC: Multimedia Kiosks for Social Services, ACTS project AC030, , Broadcom Eireann Research Ltd. HTTP://WWW.UK.INFOWIN.ORG/ACTS/RUS/TRIALS/ac030.htm

7. Trace Research and Development Center, University of Wisconsin Designing More Usable Information / Transaction Machines (Fare Vending Machines, Kiosks, ATMs), http://www.trace.wisc.edu/world/kiosks/

Hippie: A Nomadic Information System

Reinhard Oppermann, Marcus Specht, Igor Jaceniak and Ruediger Huettenhain
GMD - German National Research Center for Information Technology
Institute for Applied Information Technology (FIT-PIC)
D-53754 Sankt Augustin
{reinhard.oppermann, marcus.specht, igor.jaceniak, ruediger.huettenhain}@gmd.de
http://fit.gmd.de/hci/projects/hips

Introduction

Hippie [1] is an internet-based guide offering added value to current information facilities by supporting the multiplicity of activities during the preparation, the execution and the evaluation of a museum/fair visit[2]. The process orientation is made possible by the nomadic characteristic [2] of the system that allows the user to have access to his or her personal information space from all places independently from specific devices. The context takes into account the current location and direction of the user, his personal characteristics like knowledge and interests and the environmental conditions like physical arrangements and technical tools.

Before a visit a user can browse all exhibits, prepare tours, and mark individual hotspots. The information selected and presented is adapted to the interests, the knowledge and the presentation preferences of the user [3]. During the actual visit augmented reality components for artwork interpretation and mainly audio output complement the visual modality preoccupied by the physical environment. Furthermore position tracking and location systems in the exhibition place allow for the adaptation of hippie to the current visitor position (at home or in front of a certain exhibit). The system automatically identifies the relevant objects close to the visitor and multi-modal information presentation takes into account the specific environmental constraints for information perception in the physical context.

Two main elements for the process orientation of the system, the internet-based personal information space and the learning capability of the system of the evolving knowledge and interest of the visitor, are described below in some more detail.

Nomadic Information Systems for Individualized Process Support

Internet connectivity provides access to the information basis from all over the world. At home the user can access the system with a desktop computer with high-resolution presentations to study the site of interest and can prepare an actual visit. The visit in the exhibition is supported by a handheld/wearable computer (PDA) with wireless LAN connection. Being in the museum the user can move around and explore the environment with exhibits of particular interest for him or her. The system identifies the

[2] The prototype Hippie was developed by GMD in the context of the project "Hyperinteraction within Physical spaces" (HIPS), an EU-supported LTR project in ESPRIT I[3]. The partners of the consortium are University of Siena (co-ordinating partner), University of Edinburgh, University College Dublin, ITC, SINTEF and GMD, CB&J, and Alcatel.

current position of the visitor and updates the appropriate information proposal for the visitor who can select the proposed information presentation or proceed to another interesting exhibit where information about the exhibit is welcome. The visitors access the same information space they are already familiar with from sessions at home including own annotations and hotspots or with a tour including exhibits of a particular importance for these visitors.

Information Adaptation to User's Knowledge and Interests

The adaptive component runs a user model describing the knowledge and the interests of the user. The user model automatically evaluates the user's interaction with the system in the information space and the user's physical navigation in the museum. The adaptation to the assumed pre-knowledge is performed by avoiding redundancy and by referring to earlier presentations. The adaptation to the assumed interests of the user is provided by adaptive tours and adaptive content recommendations. If a user has selected a number of objects (exhibits) the user model identifies common attributes of the selection in terms of, e.g., artist, style or genre. In case of exceeding a rule-defined threshold for a significant interest of the visitor in this kind of artworks the system initiates a "Tip" with a user specific tour containing relevant additional artworks [4]. The same rule-based mechanism is applied for the presentation of contents about the artworks.

Fig 1: Notification of an adaptive tour proposal

Summary

Contextualised information presentation takes into account more than just the user's location [5]. A contextualised information space is defined by an information repository adapted to the location, the characteristics of the user like knowledge, interest or interaction or presentation preferences. The prototype presented in this paper supports the process of art perception at three steps, the preparation of a museum visit, the execution of the visit itself and the evaluation of the visit. The system adapts the information presentation to the evolving knowledge and interest of the visitor to enrich the benefit and the visit in terms of knowledge and enjoyment.

References

1. Oppermann, R. and M. Specht. *A nomadic Information System for Adaptive Exhibition Guidance*. in *ICHIM99, International Cultural Heritage Meeting*. 1999. Washington, D.C.:

2. Kleinrock, L., *Nomadicity: Anytime, Anywhere.* Mobile Networks and Applications, (1997). **1**(4): p. 351-357.

3. Brusilovsky, P., *Efficient Techniques of adaptive hypermedia,* in *intelligent Hypertext,* C. Nicholas and J. Mayfield, Editor. (1997), Springer: Berlin, Heidelberg, New York. p. 12-27.

4. Oppermann, R., Adaptively supported Adaptability. International Journal of Human-Computer Studies, (1994). 40: p. 544 - 472.

5. Abowd, G.D., Dey, A. K., Abowd, G., Orr, R., and Brotherton, J. *Context-awareness in wearable and ubiquitous computing.* in *1st International Symposium on Wearable Computers*. 1997. October 13-14:

Design of Global User Interfaces: Never Ending Challenge?

Aim of the study

The globalisation trend involves the question whether European industrial standards can be employed in other culture areas, or whether there are peculiarities.

The project INTOPS was carried out by the University of Kaiserslautern, aiming at making statements on the . influence of culture on the design of user interfaces for machines

Methods of the study

Colour coding tests

Task 1	Task 2
Assign one colour to one of the following operation conditions!	Please choose one variety of the three colour varieties
❍ Danger ❍ Caution ❍ Normal situation ❍ Regulation ❍ General information	1 Faded Colour 2 Intensiv Colour 3 Windows Standard

Icon matching tests

Task 3	Task 4
Please choose only one of the three possible meanings for each icon!	Please choose only one of the three possible icon for each meaning!
1 auto-mode 2 stop function 3 start function	to diagnose

Structured and focused interviews

Countries

Asian Standard

- India
- Indonesia
- China
- Korea

Western Standard

- U.S.A.
- Germany

Interviewer and tester

- Engineers
- Psychologists
- Sociologists

Interviewee and testee

- Machine operators
- Maintenance personnel

- 5 companies per country
- 5 people per country

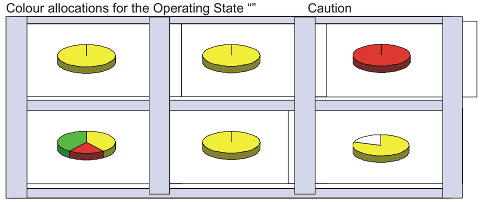
- all tests with each
- Interview with each

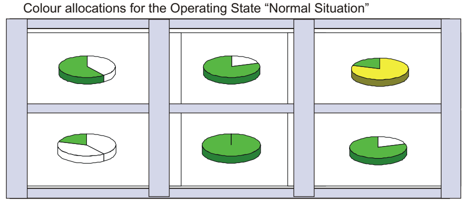
Results of the study

Results of icon matching tests

Colour allocations for the Operating State "" Caution

Colour allocations for the Operating State "Normal Situation"

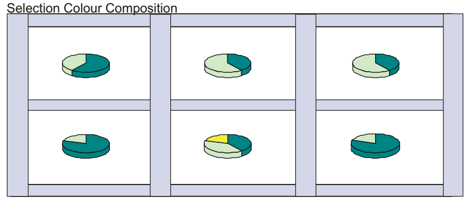




We found for different coding tests with significant results Kruskal-Wallis-Test (Alpha=0.05) and Mann-Whitney-Test (selected countries, Alpha=0.05). A homogenous allocation of colours according to . the colours red and green could be found. IEC 73 proved to be constant concerning the allocation of the with the countries examined. colour attributes for danger (red) or normal states (green)

It was striking that especially While China behaved against the general tendency for the operating state "caution" tended to be the colour yellow, laid down in the IEC 73, in China, however, 100% of the persons asked decided to have the colour red.

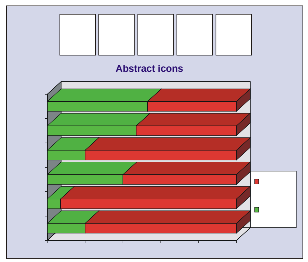
Selection Colour Composition



A test on which colour composition was also made showed another development. The increasing spread of computers lead to the development of . They are based upon the worldwide spread of the operating system of . WINDOWS

In at least 40% of all the countries investigated and often even more a colour composition that was based on the standard was preferred . This was a significant result for the screen colours, tested with Kruskal-Wallis-Test (Alpha = 0.05; p = 0.531)

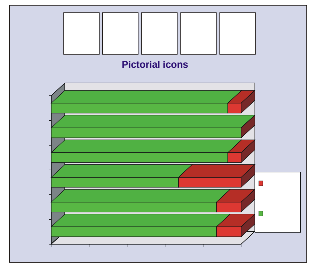
Results of icon matching tests

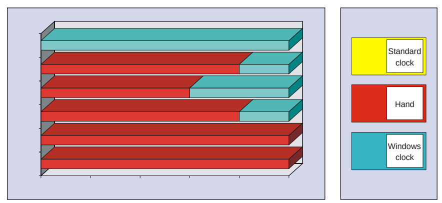


For Evaluation of test results we used the Kendall-Correlation. We found a significant Correlation (r=0.577; Alpha=0.01) for the recognisability of abstract and pictorial icons.

The results showed that the (all abstract icons used icon are DIN-Icon), standardised in Germany and often proven, were frequently misunderstood in different culture areas.

Pictorial icons , frequently being separated from the culture-specific technical development, the recognisability was much better .





If testees assign only pictoral icons to a standard situation, they choose familiar icons from the daily-use , that is an indicator for the influence from existing classification of pictorial icons from daily-use, how makes it easier to intergrate a new icon.

For the abstract icon it is much more difficult for an user to create a classification. The characteristics of abstract icons are different, especially if the criterion of familiar icons are different.

There is a general agreement on a more frequent use of pictorial icons. But the sole use of pictorial signs will not ensure an unequivocal recognisability.

Institute for Production Automation

Kerstin Röse & Detlef Zühlke

university kaiserslautern