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 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

---

[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, 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.

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

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

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.

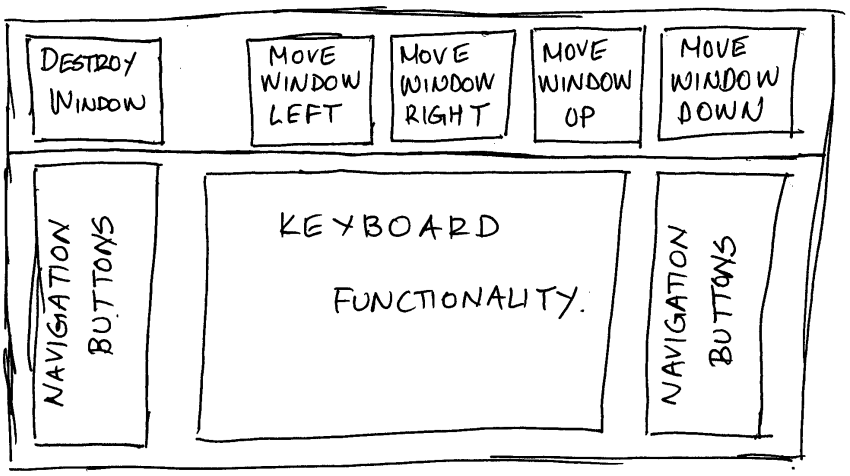

Figure 1: A paper mock-up of the virtual keyboard

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.

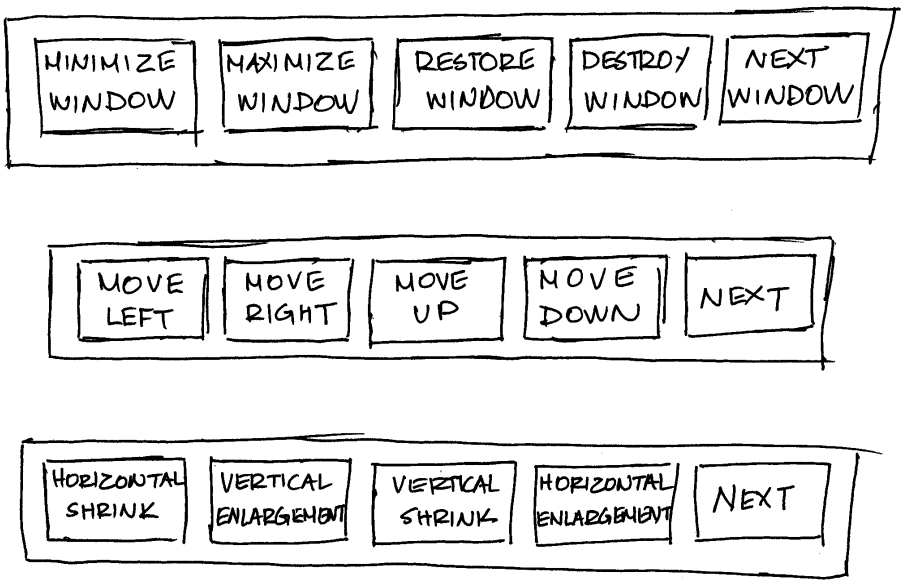

Figure 2: A paper mock-up of the toolbars for window management

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.

[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.

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. 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].

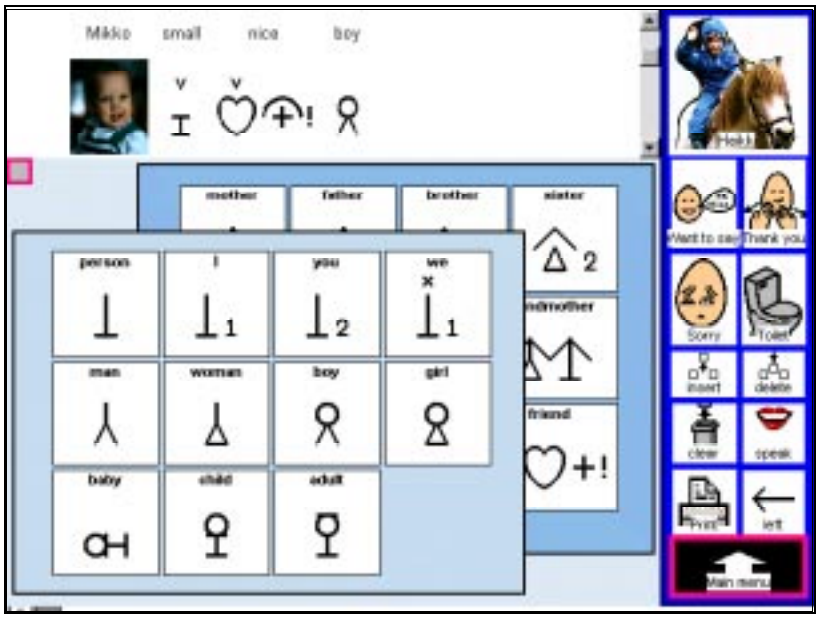

Figure 4: A screen view of a communication device, incorporating scanning, for a seven year old boy who communicates with Blissymbols.

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

---

[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).

the capability to build full-functioning emulation-based interactive systems using the scanning technique. An example of an application (namely, a 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

[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.

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.

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.