Gothenburg Studies in Informatics, Report 16, May 2000, ISSN 1400-741X

Breaking the Screen Barrier

Lars Erik Holmquist

Department of Informatics
Göteborg University, Sweden

lars.erik.holmquist@interactiveinstitute.se
www.viktoria.informatics.gu.se/play/
Tel. +46 (0) 31 773 55 33, Fax +46 (0) 31 773 55 30

Doctoral Dissertation

Abstract

This thesis is based on an important development in human-computer interface design: the move from primarily screen-based interfaces – based on the Windows-Icons-Menus-Pointer (WIMP) and Graphical Users Interfaces (GUI) paradigm developed for desktop computers – to computer interfaces which take advantage of the richness of the user's physical environment. A common thread in the thesis is the attempt to expand the user's workspace, whether that expansion is kept within the limits of the computer screen or brings the interaction to devices outside the desktop – i.e. to "break the screen barrier", figuratively or literally. The thesis consists of five papers. The first paper describes *flip zooming,* a visualization method that uses the workspace on a screen more effectively. The second paper puts flip zooming and other similar methods within a general theoretical framework, which is both descriptive and constructive. The third paper describes *WEST,* A Web Browser for Small Terminals, which was an application where flip zooming was implemented on hand-held computers. The fourth paper describes the *Hummingbird,* a mobile counterpart to desktop-based workplace awareness applications. The fifth and final paper gives a general theory for interactive systems where physical objects are used to access digital information that is not contained within the actual object. Additionally, the introduction discusses how the thesis relates to Simon's *science of the artificial,* Dahlbom's foundations for an artificial science, and *the new informatics,* the scientific discipline within which the work was performed. A spiral model of design, *Verplank's spiral,* is used to describe the research process.

Keywords
information technology, human-computer interaction,
flip zooming, mobility, awareness, token-based interaction

Language	**Number of pages**
English	142

Gothenburg Studies in Informatics
Report 16, May 2000, ISSN 1400-741X

Acknowledgements

There are many people to thank after producing a thesis work like this, and it is impossible to mention them all. The most important person has been my thesis advisor during the last three years, Bo Dahlbom, and without him, and the creation of the Viktoria Institute, the thesis would never have been written. Most of the work was performed in the PLAY research group, which thanks to the great people involved has been an excellent environment to work in. Many of the members of PLAY can be found as co-authors on the thesis papers. I am also grateful to all other co-authors and collaborators, which are acknowledged in the individual papers.

Working within the vibrant human-computer interaction research community has been truly stimulating and fun. During my short time as a researcher I have visited a lot of international conferences and research institutions, and found a great number of nice people to share ideas with, and I am grateful to all of them. During the work with the thesis I have been employed by the Viktoria Institute, the Interactive Institute, and SSKKII. Funding has been provided by SITI (through the Mobile Informatics research program), NUTEK (through the IVES and Intelligent Environments projects), KFB (through the Internet Project) and SSF (through the Interactive Institute). I am grateful for this support.

Finally, sincere thanks to Ulrika for supporting me and accepting all the long hours and time spent on travel – now we can finally go on holiday together!

Göteborg, May 2000

Lars Erik Holmquist

Contents

Front cover: The Zoom Browser; Hummingbirds; WebStickers
Back cover: Verplank’s Spiral – napkin sketch courtesy of W. Verplank

Breaking the Screen Barrier

Lars Erik Holmquist

1 Introduction

In a video presentation devised in 1981, Robert Spence and Mark Apperley presented a vision of the future office environment [38]. They saw the office of the future as a place for rich, multi-modal interaction with digital information, where users had access to a variety of input and output methods – displays of a variety of sizes, from desktop to wall-sized; gestural interaction; voice input; handwriting recognition; and so on. This was a compelling view, where users would take advantage of their whole environment to access and manage digital information, rather than being limited to a small single screen and the "point-and-click" interaction we are used to today. Although several other researchers were exploring similar avenues (e.g. the "Put-that-there" system at MIT's Architectural Machine group [7]), it is safe to say that Spence and Apperley's vision was presented well before its time. Even today, this kind of rich mix of digital and physical space is far away from being widely implemented, and is still very much at the stage of research prototypes (see e.g. [29]).

However, the reasons for why our current interaction with digital information is not as rich as that envisioned by Spence and Apperley is worth some thought. Certainly, the technology needed is quite complex – but on the other hand the authors asserted that most of their vision could be implemented with then-current technology. A more interesting "obstacle" can instead be found in what is arguably the most successful innovation in the human-computer interaction (HCI) field so far: the *Windows-Icons-Menus-Pointers* (WIMP) interaction paradigm, and the related notion of a *Graphical User Interface* (GUI), where the digital information is presented according to a *desktop metaphor*. Developed at Xerox PARC in the 1970's, and first introduced commercially with the Xerox STAR computer in 1980 [20], WIMP and GUI did not become a commercial success until the Macintosh computer was introduced by

Apple in 1984 [1], with Microsoft Windows becoming the dominant GUI several years later [12].

WIMP, GUI and the desktop metaphor has now become the totally dominant mode for interacting with computers. Most observers would agree that this is because these were very brilliant ideas. However, a problem with ideas that are as brilliant as these is that they can become very hard to look beyond, and there is a growing realization of this fact within the human-computer interaction (HCI) research community [9, 18]. In recent years we have seen an increasingly intensive search for new alternatives. There have of course been a large number of attempts to find ways to enhance user interaction while staying within the limits of WIMP and GUI. Several alternatives to the desktop metaphor have been introduced, for instance by using time rather than space as an organizing mechanism [16, 31], or by the introduction of a 3D-graphics element to take better advantage of the user's spatial perception [33]. The limited amount of space available on a desktop screen has been addressed in a number of ways, for instance by the introduction of several separate workspaces [19] or by a variety of information visualization techniques that present data more effectively [10]. Of particular interest for this thesis are those information visualization methods which attempt to "expand" the area available for showing information, through the introduction of visual distortion; notable examples include the Bi-Focal display (which was part of Spence and Apperley's vision of the future office) [38, 39] and the Perspective Wall [26].

In the years in which WIMP and GUI came to dominate the commercial side of human-computer interaction, researchers also introduced an ever-growing set of divergent approaches. *Virtual reality* promised to place users inside in a virtual world of information, where interaction would be as rich as or richer than in the "real" world [32]. *Ubiquitous computing* was based on the notion that computers would become available everywhere. It aimed to move computer usage away from the desktop-centric WIMP approach and into the user's physical environment, through the introduction of computers in various size and shapes that were specialized for different usage situations [42]. *Augmented reality* proposed to mix computational properties with real-world objects, either through a projected graphical overlay, wearable see-through computer displays, or some other computational augmentation of real-world objects [43]. *Intelligent environments* were proposed as physical environments where a variety of sensors, cameras, etc. would watch the user and where the technology embedded in the surroundings would respond according to the user's explicit or implicit wishes [11]. *Wearable computers* aimed to take the computer away from the environment and instead let it become like a piece of electronic clothing that would always be "on" and support the user at all times [27]. *Graspable interfaces* proposed providing certain functions of the GUI as physical instantiations for a more direct physical interaction [17], and in an extension of this, *tangible media* aimed to remove the border between the physical and digital world altogether, in order to "change the world itself into an interface" [21].

But however compelling these alternatives may be, there is little doubt that for many tasks, WIMP and GUI are the best modes of interaction currently available. Since they were developed specifically with the office worker in mind, typical office applications such as word processors and spreadsheets are hard to imagine functioning outside the confines of the desktop computer. Mobile technology may offer us the promise of accessing our digital information any time, anywhere, but it will probably be long before most of us can conveniently write longer texts or manage complex calculations on mobile phones or PDAs. That said, many of the uses for computers are now completely outside the office domain, and with computer technology becoming ever smaller and easier to integrate in our daily life, we will see computer technology appear in many situations we have not even dreamed of. Thus, WIMP and GUI will probably stay as the dominant interaction paradigm for most of the tasks we have traditionally associated with computer use, but there will be a myriad of complementary ways in which computers appear in our lives, and these will require radically different approaches to interaction.

This thesis spans a large part of the spectrum of interface approaches outlined above, from interaction strictly within the WIMP and GUI domain, over hand-held GUI computers, all the way to wearable computers and tangible media. The common thread is an effort to free the user from the inherent limitations of the computer screen – to "break the screen barrier", figuratively or literally. Despite the progress in recent years in both size and resolution, the typical user is still limited to a screen 17 to 19 inches in size, and with a resolution of no more than approximately 1000 by 1000 pixels. Furthermore, a computer screen is usually fixed to the same location, its mode of presentation is inherently 2-dimensional ("flat"), and it offers no provision for tactile feedback or the use of any other sensory modalities than sight. Although the computer screen is an incredibly flexible and powerful canvas for interaction design, it is clear that it is also limited in many ways. This thesis presents a few possible strategies for overcoming some of these limitations.

2 The thesis: Breaking the screen barrier

The thesis consists of five papers, four of which were published in 1999 and one which has not yet been published, but which is based in part on publications from 1997 and 1998. Apart from the required reformatting to fit the format of the thesis, the published papers are presented in unaltered form. The papers are as follows (when Holmquist is not listed as first author, authors have been listed alphabetically and/or according to affiliation):

1. Holmquist, L.E.: *Flip Zooming: Focus+Context Visualization of Linearly Ordered Discrete Visual Structures.*

   Submitted for publication.

   Based in part on the following short papers:

   Holmquist, L.E.: Focus+Context Visualization with Flip Zooming and the Zoom Browser. In *Extended Abstracts of ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*, pp. 263-264, ACM Press, 1997.

   Holmquist, L.E. and Ahlberg, C.: Flip Zooming: A Practical Focus+Context Approach to Visualizing Large Information Sets. In Smith, M.J., Salvendy, G. and Koubek, R.J. (eds.), *Design of Computing Systems: Social and Ergonomics Considerations (HCII '97),* pp. 763-766, Elsevier Science B.V., 1997.

   Holmquist, L.E. and Björk, S.: A Hierarchical Focus+Context Method for Image Browsing. In *Computer Graphics Annual Conference Series Abstracts and Applications (SIGGRAPH '98),* p. 282, ACM Press, 1998.

   Björk, S. and Holmquist, L.E.: Formative Evaluation of a Focus+Context Visualization Technique. Poster presented at *Annual Conference of the British HCI Group (HCI '98)*, Sheffield, UK, 1998.

2. Björk, S. Holmquist, L.E. and Redström, J.: *A Framework for Focus+Context Visualization.*

Abridged version in *Proceedings of IEEE Symposium on Information Visualization (InfoVis ’99),* pp. 53-56, IEEE Press, 1999. Full version in *CD-ROM Proceedings of IEEE Visualization 1999,* IEEE Press, 1999.

3. Björk, S., Holmquist, L.E., Redström, J., Bretan, I., Danielsson, R., Karlgren, J. and Franzén, K.: *WEST: A Web Browser for Small Terminals.*

   In *CHI Letters Vol 1 Issue1, Proceedings of ACM CHI Conference on User Interface Software and Technology (UIST ’99),* pp. 187-196, ACM Press, 1999.

4. Holmquist, L.E., Falk, J. and Wigström, J.: *Supporting Group Collaboration with Inter-Personal Awareness Devices.*

   Journal of Personal Technologies, 3(1-2), pp. 13-21, Springer Verlag, 1999.

5 Holmquist, L.E., Redström, J. and Ljungstrand, P.: *Token-Based Access to Digital Information.*

   In *Proceedings of First International Symposium on Handheld and Ubiquitous Computing (HUC ’99),* pp. 234-245, Springer Verlag, 1999.

The theme of the thesis is “breaking the screen barrier”, i.e. to somehow overcome the inherent limitations of the computer screen. The work started out within the WIMP / GUI domain with the intention of expanding the workspace available to a computer user, through the introduction of a means to show more information on a desktop computer screen. This resulted in the so-called *flip zooming* visualization technique presented in the first paper of the thesis, *Flip Zooming: Focus+Context Visualization of Linearly Ordered Discrete Visual Structures.* The technique could be used to visualize documents, images, and other data sets. Flip zooming was designed with a traditional desktop computer in mind, but represents a quite different approach compared to windowing systems and the desktop metaphor. By virtually expanding the available workspace to become much larger than the physical screen, it was a first step towards “breaking the screen barrier”. Subsequent work with flip zooming resulted in the more general framework for focus+context visualization presented in the second paper, *A Framework for Focus+Context Visualization,* where a formal method for describing and constructing focus+context visualizations was introduced. This work, though based in the GUI domain, is possible to generalize to other interaction approaches, e.g. 3D-environments.

Furthermore, we soon realized that the flip zooming technique could be applied to other devices apart from WIMP computers, such as mobile computers with wireless internet connections. This resulted in *WEST: A Web Browser for Small Terminals*, which is also the title of the third paper. Although GUI based, the input and output capabilities as well as the usage conditions of a mobile terminal are drastically different from that of a desktop computer. This proved to require some major changes in our approach to interface design, which are partly documented in the paper and which we have continued to explore in later works (e.g. [3]). Mobile computing, whether GUI-based or built on other approaches, is likely to be a major area for future HCI research.

In parallel with the work of flip zooming, we also explored other, quite different, approaches to human-computer interaction. *The Hummingbird* was a specialized wearable computer, which did not in any way resemble a desktop computer, and it had no GUI whatsoever. The Hummingbird and some preliminary evaluations are discussed in the fourth paper, *Supporting Group Collaboration with Inter-Personal Awareness Devices*. The purpose of the Hummingbird was quite similar to many desktop-based awareness systems, but by being completely mobile and independent of any infrastructure, it acknowledged the fact that users spend much of their time away from the desktop. Thus Hummingbirds literally “broke the screen barrier” by moving the interaction away from the desktop computer completely. This work has continued with the development of a second generation of Hummingbird prototypes and a variety of evaluations (e.g. [41]).

Finally, our work with the *WebStickers* system [24] led to the fifth and final paper. WebStickers let users couple everyday physical objects with digital information, in a fashion similar to some systems for augmented reality. In the paper, *Token-Based Access to Digital Information*, we used the lessons learned from the WebStickers system to draw some general conclusions about proposed interaction paradigms such as tangible media and graspable interfaces. This paper generalizes the interaction with computers to such an extent that screen output is just one of a variety of possible interaction methods, so that the user’s whole physical space here becomes an arena for accessing and ordering digital information – the “screen barrier” has truly been broken.

Taken together, the papers also represent a growing realization that the notion of “context” as it has been used in research on graphical user interfaces is very limited, since it only acknowledges interaction taking place on a computer screen. Instead, “context” in human-computer interaction should in reality also include the whole physical world within which the user interacts. This is consistent with the many proposed alternatives to WIMP and GUI that were outlined in the introduction. At the same time, the thesis also reflects the fact that WIMP and GUI will be with us for yet some time and that it is still very worthwhile to explore enhancements and augmentations within that domain, a notion that is sometimes easy to forget in the rush towards the various tangible, mobile, ubiquitous and wearable alternatives. By playing off different interaction paradigms ranging from WIMP and GUI to augmented reality, tangible interfaces and wearable computing, the papers in the thesis thus explore how human-computer interaction can “break the screen barrier” both in a figurative and a literal sense.

3 Research method

This thesis is a work in the Swedish scientific discipline called *informatics,* or more specifically the *new informatics* as proposed by Bo Dahlbom in 1996 [13]. The Swedish discipline of informatics could be described as “applied computer science”, and has its roots in information systems research. The subject matter of the new informatics is *information technology (IT) use.* However, the new informatics is not focused only on studying the use of IT – it is also very much interested in changing and improving the use of information technology. In other words, it is a *design oriented discipline* [13, p. 29]. According to Dahlbom, “whatever we do with our discipline (...) we should protect our design interest” [13, p. 30], stressing the importance of the discipline’s active involvement in and contribution to the development of information technology and its use. This sentiment is in line with the work in this thesis, which has been very much concerned with the development of novel IT artifacts and the exploration of their use.

The new informatics’ strong interest in design follows from the notion that it is an *artificial science,* i.e. one that is concerned with objects created by man, as opposed to the *natural sciences,* which study things in nature. The idea of an artificial science was presented by Herbert Simon in a series of lectures in 1968, which were published in book form in 1969 [34], and revised several times, most recently in 1996 [35]. Simon argued that there is a need to acknowledge the importance of design in the engineering disciplines, rather than to have them fall into the trap of mimicking the methodology of the natural sciences in an effort to gain scientific credibility. Influenced by then-current research in artificial intelligence, Simon also proposed a theory of design, where the design activity takes the form of a search for the best among a certain set of available alternatives. However, a major problem seems to be that Simon's design theory is unable to capture the creative, intuitive and accidental elements in the design process, which often can mean the difference between a brilliant design and one that is merely satisfactory.

Dahlbom has addressed part of this problem in a work that is a direct response to Simon [14]. Although an enthusiastic supporter of the general idea of artificial sciences, Dahlbom presented several objections to Simon, having mainly to do with Simon's definition and theory of design and its role in the artificial sciences [14, pp. 5-6]. Dahlbom then proceeds to propose a set of foundations for an artificial science, four of which are directly inspired by Simon's work, and four which also are inspired by Simon but are more specifically based on the foundations of the natural sciences. I will summarize the eight points below; for more details see [14, pp. 6-9]. (Dahlbom's text is in italics; the summaries are mine.)

1. *Artifacts are designed rather than described,* i.e. artificial science studies possibilities rather than the already realized; also, it insists on concrete realization as a way to make sure something is really possible.

2. *Technology in use rather than design practice,* i.e. the research will be design oriented from a user perspective, considering improvements in use quality rather than product quality per se.

3. *Artifacts have quality rather than functionality,* i.e. we will have to go beyond thinking of the relations between people and technology just in terms of "use", and introduce dimensions such as aesthetics, symbolism, ethics and politics.

4. *Artificial science is normative rather than objective,* i.e. since artificial science is concerned with more than an artifact's functionality, it

involves the idea of the “good life”, and thus it goes beyond purely objective concerns.

5. *Artifacts are accidental rather than essential*, i.e. rather than being nicely broken down into simple principles, artificial laws are local design solutions rather than general principles, and the artificial world is haphazard and provisional.
6. *Artifacts are constructed rather than documented*, i.e. results are only judged on the basis of successful construction – it is the quality of the technology that matters, not the documentation.
7. *Artificial science has heuristics rather than methods*, i.e. whereas the emphasis on methods turn natural science into a bureaucratic administration of ideas, creativity takes the center stage in artificial science, requiring a reliance on heuristic rules of thumb, intuition and tacit knowledge, experience and tinkering.
8. *Artificial science is engaged rather than disinterested*, i.e. the artificial sciences are not interested in objective detachment but in interacting with artifacts, and values play an important role in how one chooses what to work with, making artificial sciences more openly politicized.

The details of these points should of course be open for debate and discussion, but they are all quite easy to accept for this author, and as we shall see we will have no problem placing the work presented in this thesis within this framework. But what about the *process* of designing new artifacts? If we want to produce innovations in the field of information technology and IT use, what is the process we should follow? Dahlbom says that “artificial science is not a theoretical study of the design of concrete artifacts, but a systematic, institutionalized form of such design activity with the ambition to improve the world of artifacts” [14, p. 5]. Several of Dahlbom’s principles stress the importance of practical and use-oriented design, but fail to give readers much help in how this design actually happens.

Going back to Simon, design is “... concerned with how things ought to be, with devising artifacts to attain goals.” ([35, p. 114]). This sounds very simple: we take a look at the world we live in, we figure out how it might become better, and we design an artifact which takes us to this new, better world. There are indeed good methods to help us understand the world we live in; ethnographical studies, economical models, etc. There is also an abundance of information technology that could help us in attaining our goals – not just desktop computers, but also mobile devices, embedded chips, smart materials, sensors and actuators, and so on. Thus, the stage seems well set for the new informatics researcher: armed with a well-founded knowledge of the world and a set of novel information technology, he or she can easily proceed to change the world into a better place!

But as the reader probably already has noticed, there is one thing missing: the goal, that better place the world should be. How we find out what this goal is seems to be the one crucial thing missing from Simon's definition of design. Dahlbom addresses that this is a problem, in particular by bringing in other dimensions such as aesthetics and politics, thus implying that we should strive for goals that increases the "happiness" of the user, but of course in the same breath acknowledging that happiness is a very much a "rubber concept" [14, p. 7]. This does not really take us very much further: yes, we should strive to find "good" goals, but what are they and how do we find them?

One approach can be found within the first strand of the new informatics to define its own research agenda: *mobile informatics* [23]. Mobile informatics is aimed at inventing new IT use in mobile settings through interdisciplinary collaboration, and identifies two methods for idea generation (i.e. goal definition): idea generation informed by studies of current practice; and technologically informed idea generation. But again, this does not help us in understanding the design process – that we know what the world is like, and what technology can currently do, does not guarantee that we will come up with innovative (let alone "good") uses for technology. In fact, this is still very close to Simon's definition of design (above). However, a hint of what the process might entail is found in the emphasis on practical implementation of artifacts, which is identified as being necessary for finding limitations and possibilities: "The very construction of the IT artifacts will (...) give rise to new insights (and) form the IT artifact being implemented" [23, p. 207].

An approach which has been described in detail within the framework of mobile informatics, is that of so-called *scalability through cultivation* [2]. Here, the goal is not to radically change a work situation through the introduction of novel IT use, but to support the current work practice through "guided evolution", addressing only the badly functioning parts. In practice, this is carried out by first doing a study of a workplace (inspired by ethnographical methods), and then devising design proposals based on the results of that study. The design proposals are then presented to the workers for comments. This is reminiscent to the method known as *participatory design,* where the prospective users are directly involved in the construction of computer systems [28].

The case at hand concerned an order packaging department, but the methodology should be possible to generalize to many other situations. The most obvious strength in this approach is that since the resulting design proposal is based on a study of the workplace, rather than being a "flash of inspiration" thought up in isolation from the actual work practice, it should have a greater chance of addressing the important problems. Also, since the approach stresses the importance of "cultivation" rather than radical change, proposed designs should have a greater chance of being implemented without disrupting current work practice.

However, the fact that a proposal is based on a study does not guarantee that it is the best possible proposal, not even that it is a good one! The study and design proposals given in the paper can be taken as an example. For instance, the study shows that physical work orders (pieces of paper) that are taken from a communal notice board are used as a coordination mechanism. The proposed design solution *removes* the physical work orders and replaces them with a system consisting of a large computer screen (replacing the notice board) and a set of networked hand-held computers (containing the work orders), one for each worker. The authors claim that using work orders on paper is less efficient than using computers, but in fact there are several studies that show how physical artifacts such as paper are an important support in many work processes, even those which rely heavily on computer support (e.g. [25]). Although removing such a mechanism and replacing it with computer technology might in some cases make work more effective, it might just as well spell disaster in the current work practice! Until the proposed design has been implemented it is impossible to know what the effect might be. In any case, an alternative design proposal which integrated the existing coordination mechanism of the physical work order with some kind of computer support might have presented a better solution to many of the problems identified in the study.

The reason for bringing up this example is not to criticize it in depth, but to point out that studies, no matter how well done, are always very much open to interpretation. Although the facts may be correct, this does not guarantee that the conclusions are. Even more importantly, the quality of design proposals that are based on such a study is not in any way certain; one person might have a brilliant idea based on a study, whereas another might come up with something which does not improve the situation in any way, and perhaps even makes it worse. The only way to know is to carry through with the "cultivation" – to really implement the proposal and see what happens, and then make further adjustments based on that. Thus, until a proposal resulting from an observation has been taken back to the workplace and implemented, there is really no way of judging its quality; just asking the workers if they like the proposal will certainly not be enough.

Another objection to the cultivation approach is that it only promotes incremental change, not radical innovation. This is probably as it should be; if we want to cultivate a current work practice, and improve it without destroying it, cultivation is a reasonable approach. However, if we want to introduce a new work practice, or if we are interested in true innovation, cultivation might not work very well. Any radical departure from the current work practice will then seem as a threat and quite probably be met with opposition. Furthermore, many truly ground-breaking innovations – the telephone, the car, the internet, and so on – are not specific but general in their use and effects, and thus not rooted in any easily studied practice. In fact, it is quite hard to see how such innovations could arise from any kind of workplace study.

Returning to the work in this thesis, rather than starting with studies of work practice, it has been based on a practical approach to design, fitting well with Dahlbom's foundations for an artificial science. We have focused on producing *innovations* rather than incremental improvements. But innovations have no value in themselves – they need to be proven useful; to be doable, to actually have a place in the real world. The only way of doing this is by practical implementation. By not only dreaming about innovative artifacts but actually *creating* them, we will find out much more about them than is possible by staying on a purely conceptual level. When an artifact is actually built and tried in the real world, rather than just presented as a design suggestion, one will often find it to be a very different beast than one thought. Not only does the physical implementation of artifacts tend to turn up many unexpected problems; more interesting for the new informatics researcher is that the use of a novel artifact is never uncomplicated. Unexpected things happen when people get their hands on artifacts – things the developer could never have foreseen. This element of uncertainty, the fact that when real people are allowed to use an artifact they will offer both criticisms and suggestions, should be at the center of new informatics research. This response in turn should be channeled back into the design of the artifact, so that it may change accordingly.

Thus, the process of doing this research can not be broken down cleanly in “before-after” situations, or in “states” and “goals”, because the goals are not known – or at least not very well specified – when the work begins. When approached from such a practical perspective, the reason for the lack of a clear methodology to find design goals is probably this: the goal in any design process is very much a moving target, and design in itself is a crucial factor in making it move. Rather than being something which is defined at the start of a project and reached at the end, the goal of most design processes changes continually as the design evolves. This is especially true when design takes place within a research context, where the interest often lies as much or more in finding questions as in answering them. Thus, there is a need to complement the new informatics with a more detailed theory of the design process, which manages to take into account Dahlbom’s foundations of an artificial science.

The process of designing computer software might offer a useful parallell.[1] While programming originally was an informal practice undertaken by a single person or very small teams, it has now reached the stage where it is often a time-critical effort involving hundreds or even thousands of people. Several models have been proposed and used to steer the software development process. An early model was the *waterfall model* [5], where development is performed in a series of well-defined steps, where the software is “frozen” at each stage. This is very similar to well-functioning methods for developing complex hardware systems, where each component is specified and produced according to certain criteria; however, hardware is very different from software. In software, “once the drawings and models (programs) are complete, the the final product exists” [4, p. 30]. An analogy in [4] to house building versus sculpting is enlightening: whereas a house is built with a good understanding of the requirements, and modifications are restricted to cosmetic and minor items, a sculpture is much less rigid, since clay can be added and subtracted during the whole process. In sculpting, the process of making the sculpture is part of finding out what the sculpture

1. The following two paragraphs on software development is based on [36]; quotations are not first-hand, but taken from that text.

will look like, which fits well with our notion of the design process in new informatics research.

Several models that try to capture this iterative nature of software design have been proposed, with Boehm's *spiral model* perhaps the most well known [6]. In this model, a software project will start with a set of requirements, but these are not cast in stone; instead, they will change and develop as the development continues. Through a series of risk analysis, prototyping, and verification cycles, a piece of working software is constructed. The difference between this and the waterfall model is that software is never "frozen"; instead, it evolves, with changes and amendments happening over time.

Another spiral model, which fits very well with the work in this thesis, is what we will term *Verplank's Spiral,* devised by William Verplank when working at Interval Research.[1] This model is not limited to software development but was developed to describe the general process of developing marketable products. The model stresses the flexible nature of design even more than Boehm's Spiral, since rather than starting out with a set of requirements, it start with something as unspecific as a "hunch"! Also, whereas the models for software development are meant to steer or guide the work, Verplank's Spiral should be considered as more of a descriptive tool.

Verplank's model (**Figure 1**) is placed in a continuum where the vertical axis denotes the dimension of *paradigms* versus *industries.* In this model, a project typically starts out with a *hunch* – a vague notion of what to do. This leads to a *hack* – a first, primitive technical demonstrator of some kind. The hack makes it possible to *try* if the hunch is valid, and this in turn leads to an *idea.* The idea leads to one or more *designs;* this is the place in the spiral where several alternative avenues present themselves, since it might be possible to devise more than one design from the same idea. The designs are then fashioned into *prototypes* – working instantiations of the design. The prototypes can then be *tested,* and this in turn leads to a set of *principles* arising from the tests. These principles can then be fashioned into *plans,* which are specific enough to be used for *production* of actual products which reach the *market.*

1. The material on Verplank's Spiral is based on personal communication, William Verplank, Elsinore, Denmark, April 14, 2000. See also [15].

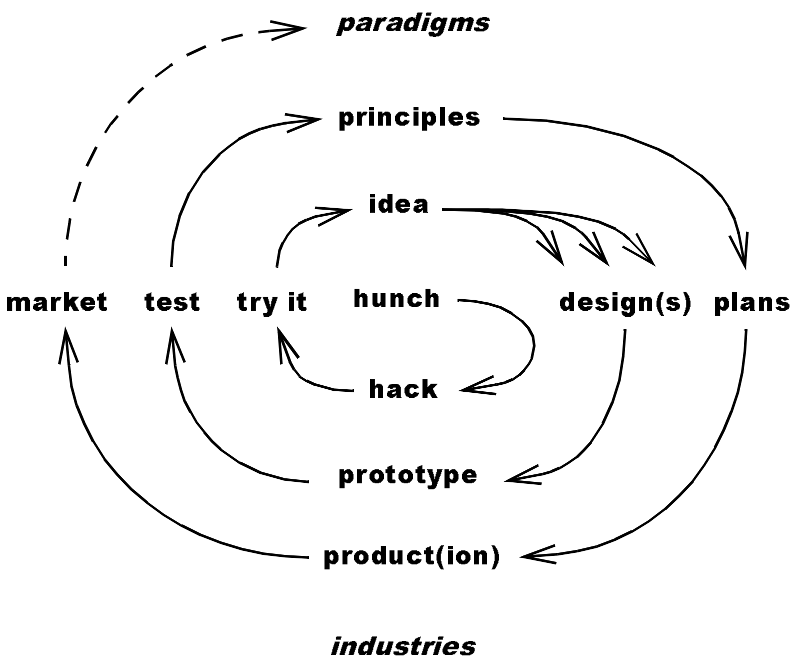


Fig. 1. Verplank's Spiral.

Finally, if the product is successful it might give rise to a new *paradigm*, that might even become an integral part of out lives (this would presumably happen quite rarely, and I have chosen a dashed line to indicate this, and also in acknowledgement that it is not absolutely clear to me whether Verplank actually meant paradigms to be an actual design goal as well as an underlying dimension). It is easy to see how all major technological inventions, from the printing press to aeroplanes, from television sets to the desktop computer, can be made to fit into this cycle.

The spiral is of most interest here because it allows us to find the place of various activities in research and development and to relate those to each other within the spiral. For instance, the so-called "demo-or-die" approach (most famously embodied by the MIT Media Lab [8]) is an example of an activity which stays very much within the "hunch" and "hack" domain. By creating an environment where hunches are encouraged and can be nurtured into hacks, so-called "demos", such an institution can produce a wide variety of exciting new ideas which challenge the status quo. On the other hand, one might question the value of some of those ideas: if they do not reach further out into the spiral, it is hard to see how they might have a real impact on everyday life, especially if there are no resulting general principles to take away from them. But as long as the main purpose is to produce exciting innovations, having hunches and doing hacks is quite sufficient.

Similarly, an institution that concentrates on testing, such as for instance a usability lab, would be working very much in the domain of tests and coming up with general principles as a result. In this part of the spiral, the important thing is to examine existing prototypes and subject them to tests which are rigorous and wide-ranging enough to come up with general rules as to what works and what does not. This activity (as the spiral nicely illustrates) is much closer to developing plans that in turn can become actual products for the market, and might thus seem to be more relevant from the perspective of a commercial enterprise. On the other hand, by concentrating on evaluating and refining existing prototypes, the likelihood of producing ground-breaking innovations is probably much smaller than when working closer to the centre of the spiral.

Any truly successful design would thus start at the center of the spiral and move all the way out. But this is not the same as saying that the same person or team has to do everything! Although the occasional lone inventor might have been able to take a hunch all the way to a marketable product, any serious enterprise would certainly require several different specialists in different areas to go through with the whole process. This would seem to mean that there is nothing wrong with one group producing hacks, another doing designs and prototypes based on the resulting ideas, a third testing them and coming up with principles, etc. But having different parts of the design process separated also leads to problems, and decisions made early in the process might be hard to affect in the later stages, sometimes leading to products which are not as good as they should be. Donald Norman suggested *human-centred development* as a solution: by letting a user-centred perspective be a part of the process from the very start, he argues that better, more usable and less complex products will appear as a result [30]. This would presumably mean that the notion of a user would be present all the way from the first hunch, rather than being something which is added at the last minute when a product turns out to be too complex or user-unfriendly.

But it is also worth noting that Verplank's Spiral aims to describe the process of producing *products*; it is not certain that it is directly transferable to scientific research. Whereas a commercial company is mainly focused on producing artifacts that can be sold with a profit, a scientific institution is usually more interested in coming up with principles and paradigms. In fact, scientists might want to skip the product and market phases altogether and generate principles that are powerful enough to become paradigms in their own right. We can find many examples of this in the natural sciences. For instance, the "Big Bang" theory of the origin of the universe was certainly not developed with the intention of turning it into a successful product, but it has still entered into our consciousness and fundamentally altered the way we think about the world.

Can the same be said for the science of the artificial – can we cut products and markets out of the loop and still produce good scientific works? Probably not. A science of the artificial is concerned with the design of artifacts, and these artifacts are meaningless unless they enter into a relationship with a human being. If we define a "product" as an artifact which is produced with the intention that someone will want to use it, then artificial science is definitely concerned with products, whether we like it or not. The artifacts we produce may not be mass-market items; indeed, they might not satisfy a single person, not even the designer herself. But the perspective that someone, somewhere must interact with them seems to be fundamental to producing successful work in an artificial science, and thus also in the new informatics. This also seems to agree very well with Dahlbom's eight foundations of an artificial science. In the following we have therefore chosen to adopt Dahlbom's eight points as a general framework for the work in this thesis, and Verplank's Spiral as a tool to describe the type of work performed in the individual papers.

4 Relating the thesis to the research method

If we now return to Dahlbom's eight foundations, we can examine how well the thesis material fits within his definition of an artificial science. This work has relied on a practical approach to information technology, so that by designing and trying out novel designs, we have gained a better understanding of what is possible (point 1). It has been permeated by a user-centred perspective, so that each artifact has been constructed with an idea of use and users rather than being examples of exciting and advanced technology (point 2). The resulting artifacts have been evaluated not according to how efficiently they solve a particular problem, but how the user experiences them in practice (point 3). Because of this, there has been no objective way of telling whether a design has been successful; rather, this has been done on the grounds of experience and user reactions (point 4). Although some of the artifacts were the results of well-thought-out research agendas, whereas others started as pure hunches, the process in arriving at the final results has in all cases been one of experimentation, staying with some solutions and discarding others (point 5). In the empirical work (papers one, three and four), every one of the artifacts has been a working instantiation of an idea; in the theoretical work (papers two and five) we have aimed to construct useful and productive frameworks rather than theory for its own sake (point 6). Creativity and accidents has been key ingredients in arriving at many of the final results, just like it is in all design work (point 7). And finally, the work has addressed key issues such as privacy, information overload, and how we can make information technology easier to use (point 8).

The general framework of an artificial science thus fits well with the work. What about the design process and Verplank's Spiral? Here, we must look at each paper individually. The first paper, *Flip Zooming: Focus+Context Visualization of Linearly Ordered Discrete Visual Structures,* started with the hunch that something could be done about screen real estate, and that the way people handle documents in the physical world could be used as a source of inspiration. A series of hacks led to the invention of the flip zooming visualization technique, which is the *idea* that is the main contribution of the paper. The idea of flip zooming then resulted in several designs and prototypes, three of which are described in the paper – The Zoom Browser, The Flip Zooming Image Browser, and The Hierarchical Image Browser. (Many more designs and prototypes have been produced based on flip zooming, but they are for the most part outside the scope of this thesis.) But the prototypes were not developed in parallel; the design of each one was influenced by the experience with the previous prototypes, stressing the iterative nature of the design process.

The second paper, *A Framework for Focus+Context Visualization,* built directly on the previous work to come up with some general *princi-*

ples for focus+context visualization. By looking at the many designs and prototypes produced both by ourselves and others in this area, we could find some common factors, and express these in a formal way. Thus the main contribution of this paper is the foundations for a formal system that might eventually be powerful enough to both describe existing focus+context visualization techniques, and produce novel ones.

The third paper, *WEST: A Web Browser for Small Terminals,* took the flip zooming idea all the way back to the hunch stage. The hunch was that flip zooming might not only be useful on desktop computers; it could also be applied to the new breed of handheld computers that were becoming increasingly popular. However, it also soon became evident that flip zooming in itself was not powerful enough to solve the task at hand, in this case presenting a clear view of an ordinary web page on a device with limited processing power and memory, and with a display limited to only 160x160 pixels. But by bringing in a number of other techniques, including proxy pre-processing and text summarization, we could solve the problem, and present the WEST *prototype,* which is the main contribution of this paper. This prototype has attracted enough interest for us to think that it might become a viable commercial product, and WEST could become one of the results in this thesis that travels all the way through the spiral and out to a commercial market.

The fourth paper, *Supporting Group Collaboration with Inter-Personal Awareness Devices,* was another return to the hunch stage, but this time without flip zooming as companion. “What would happen”, the hunch went, “if we had a set of devices which would *hum* if they got close to each other”? This turned out to be a not very original hunch (see e.g. [22]), but the subsequent process turned the hunch into something quite different and much more interesting. We produced a hack which proved that this was technically feasible, and eventually came up with a design which we called the *Hummingbird.* Hummingbirds are indeed devices that hum when they get close enough to each other; but the novel element is that we chose to use them as a support for group collaboration rather than an initiator of chance encounters (which has been the goal of most similar commercial systems). Thus Hummingbirds perform a similar task to many desktop-based awareness applications (e.g. [40]) but do so completely outside the scope of WIMP and GUI. This *idea,* of moving awareness information away from the desktop and to the user, is the main contribution of this paper. The Hummingbirds is the second result of this thesis which has attracted commercial interest, and plans are currently being put into place to turn it into a product.

The fifth paper, *Token-Based Access to Digital Information*, is like the second paper mainly concerned with coming up with *principles*. Through our own implementation of the WebStickers system [24], and through the observation of several other similar systems which allow users to couple digital information with physical objects, we were able to come up with some general results for classifying such systems. The schema of *containers, tokens* and *tools,* which aims to capture three significantly different ways in which physical objects can be associated with digital information, is the main contribution of this fifth and final paper.

We can thus see that the results of the five papers in this thesis occupy a variety of different places in Verplank's Spiral. Some stay very close to the hunches and hacks, with the main contribution being an idea arising from this, whereas others are more concerned with general principles. And a few of the results might have the strength to travel all the way out of the spiral and into the marketplace – whether this will actually happen is still too early to tell. In any case, we believe that for scientific work in an artificial science to be successful, it is important to acknowledge all parts of the spiral, although one may choose to focus one's work only at a certain section. The work in this thesis has certainly benefitted from hunches as well as tests, and produced ideas as well as prototypes, designs as well as principles.

But is this work, with its focus on implementation and prototyping, really scientific research, and not "just" product development? The Merriam-Webster's Collegiate Dictionary (online edition) offers several definitions of research, including: "investigation or experimentation aimed at the discovery and interpretation of facts, revision of accepted theories or laws in the light of new facts, or practical application of such new or revised theories or laws". The work in this thesis has been just such an investigation, with both practical and theoretical components. That some products may indeed come out of the work is true. However, that the nature of the work has been very different from that in a commercial R&D unit should also be clear. It has been very much an exploratory effort, with the intention of testing the limits of the human-computer interface rather than solving any specific problem. Any potential products that have resulted from this have been side effects rather than the main goal. Instead, the main results of the work has been an increased general understanding of how we can enhance human-computer interaction within or outside the WIMP / GUI paradigms.

5 Conclusion

In this thesis, I have explored a variety of different approaches to make human-computer interaction “break the screen barrier” – to make interacting with computers a richer experience, not limited to the screen of the desktop computer. The work has taken the form of a practical exploration of the possibilities of modern information technology, so that by constructing artifacts and seeing if they work, ideas, prototypes, principles (and perhaps even products) has emerged. This empirical approach to IT research seems to be very much in agreement with the foundations of an artificial science put forth by Bo Dahlbom [14], and also fits well within the scientific agenda of the new informatics [13]. However, it is different from approaches that stress studies and cultivation of current work practice (e.g. [2]), since this work has taken the form of exploratory design of artifacts rather than improvements based in current practice.

Increased understanding of the human-computer interface is today more important than ever before. Spence and Apperley’s vision of the office of the future has still not been realized, although many of the required puzzle pieces have been put in place during the last two decades of research – some of them might perhaps even be found in this thesis. But the use of computers is no longer limited to the office, or even to work. Whereas computers where once specialized and expensive pieces of equipment, computers can now be found almost everywhere. Most families in Sweden have home computers, and electronic entertainment systems are available which approach or surpass the power of current PCs. And this is not all: there are computers everywhere – in cars, dishwashers, TV sets, watches... We are literally surrounded by computers!

Computers can now be made small and inexpensive enough to put inside everyday objects such as furniture and books, jewelry and clothing. Computation has the power to permeate our entire lives, to seep into the very fabric of our existence, just like electric power has managed to do during the last one hundred years. Computers will appear in situations we never thought possible, aiding, entertaining and comforting us without us even knowing they are there – computers will become invisible. Yet at the same time computers will stay the same: the desktop will not go away. Word processing and calculations will take place on desktop computers until a more powerful alternative presents itself and supplants the desktop, much like computers replaced the typewriter and adding machine that came before them. And there is still much to be explored in the mix between different approaches: on the boundaries between stationary computers and mobile devices, between the digital space and the physical, between powerful number-crunching servers and little things that think. This area, when we have truly "broken the screen barrier", is where this author believes some of the major innovations in human-computer interaction are still about to happen.

6 References

1. Apple Computer. *Macintosh Human Interface Guidelines*. Reading, MA: Addison-Wesley, 1992.
2. Bergqvist, J. and Dahlberg, P. Scalability Through Cultivation. *Scandinavian Journal of Information Systems*, Vol 11, 2000.
3. Björk, S., Holmquist, L. E., Ljungstrand, P. and Redström, J. Providing Effective Interaction on Small Screens with PowerView. In *Extended Abstracts of ACM SIGCHI Conference of Human Factors in Computing Systems (CHI 2000)*, ACM Press, 2000.
4. Blum, B.I. *Software Engineering: A Holistic View,* Oxford University Press, 1992.
5. Boehm, B.W. *Software Engineering Economics,* Englewood Cliffs, NJ: Prentice-Hall, 1981.
6. Boehm, B.W. A Spiral Model of Software Development and Enhancement. *IEEE Computer,* May, 1988.
7. Bolt, R.A. Put-That-There: Voice and Gesture at the Graphics Interface. In *Proceedings of Computer Graphics Annual Conference Series (SIGGRAPH '80),* ACM Press, 1980.
8. Brand, S. *The Media Lab: Inventing the Future at M.I.T.* New York, NY: Viking Penguin, 1987.
9. Buxton, W. *Out From Behind the Glass and the Outside-In Squeeze.* Invited speech at ACM SIGCHI Conference of Human Factors in Computing Systems (CHI '97), Atlanta, Georgia, USA, 1997.
10. Card, S.K., Robertson, G.G. and Mackinlay, J.D. The Information Visualizer, an Information Workspace. In *Proceedings of the ACM SIGCHI Conference of Human Factors in Computing Systems (CHI '91),* pp. 181-186, ACM Press, 1991.
11. Coen, M., (ed.). *Intelligent Environments.* AAAI Spring Symposia Series, Technical Report SS-98-02, AAAI Press, 1998.
12. Cringely, R.X. *Accidental Empires: How the Boys of Silicon Valley Make Their Millions, Battle Foreign Competition, and Still Can't Get a Date.* 2nd ed. HarperBusiness, 1996.
13. Dahlbom, B. The New Informatics. In Ljungberg, F. (ed.), *Informatics in the Next Millennium,* pp. 15-35, Lund: Studentlitteratur, 1999. Originally published in *Scandinavian Journal of Information Systems,* 8(2), pp. 29-48, 1996.
14. Dahlbom, B. The Idea of an Artificial Science. In Dahlbom, B., Beckman, S. and Nilsson, G.B., *Artifacts and Artificial Science,* pp. 2-15, August 1999. Available from: http://www.informatik.gu.se/~dahlbom/

15. Davenport, G. Holmquist, L.E., Thomas, M. Fun: A Condition of Creative Research. *IEEE Multimedia,* July-September, 1998.

16. Fertig, S., Freeman, E., and Gelernter, D. Lifestreams: An Alternative to the Desktop Metaphor. In *ACM SIGCHI Conference on Human Factors in Computing Systems Conference Companion (CHI '96),* pp. 410 - 411, ACM Press, 1996.

17. Fitzmaurize, G.W., Ishii, H. and Buxton, B. Bricks: Laying the Foundations for Graspable User Interfaces. In *Proceedings of the ACM SIGCHI Conference of Human Factors in Computing Systems (CHI '95),* pp. 442-449, ACM Press, 1995.

18. Gentner, D. and Nielsen, J. The Anti-Mac Interface. *Communications of the ACM,* 39(8), pp. 70 - 82, 1996.

19. Henderson, D.A. and Card, S. Rooms: The Use of Multiple Virtual Workspaces to Reduce Space Contention in a Window-based Graphical User Interface. *ACM Transactions on Graphics,* 5(3), pp. 211 - 243, 1986.

20. Hiltzik, M.A. *Dealers of Lightning: Xerox Parc and the Dawn of the Computer Age.* HarperBusiness, 1999.

21. Ishii, H. and Ullmer, B. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In *Proceedings of the ACM SIGCHI Conference of Human Factors in Computing Systems (CHI '97),* pp. 234-241, ACM Press, 1997.

22. Kahney, L. *"Hi, Do You Beam Here Often?",* Wired News (Web-based news service), March 25, 2000. URL: http://www.wired.com/news/technology/0,1282,35090,00.html

23. Ljungberg, F., Dahlbom, B., Fagrell, H., Bergqvist, M. and Ljungstrand, P. Innovation of New IT Use: Combining Approaches and Perspectives in R&D Projects. In *Proceedings of the Fifth Biennial Participatory Design conference,* pp. 203-210, ACM Press, 1998.

24. Ljungstrand, P., Redström, J. and Holmquist, L.E. WebStickers: Using Physical Tokens to Access, Manage and Share Bookmarks to the Web. In *Proceeding of Designing Augmented Reality Environments (DARE 2000),* pp. 23-31, ACM Press, 2000.

25. Mackay, W.E., Fayard, A-L., Frobért, L. and Médini, L. Reinventing the Familiar: Exploring an Augmented Reality Control for Air Traffic Control. In *Proceedings of the ACM SIGCHI Conference of Human Factors in Computing Systems (CHI '98),* pp. 558-565, ACM Press, 1998.

26. Mackinlay, J. D., Robertson, G. G., Card, S. K, The Perspective Wall: Detail and Context Smoothly Integrated. *Proceedings of the ACM SIGCHI Conference of Human Factors in Computing Systems (CHI '91),* pp. 173-179, ACM Press, 1991.

27. Mann, S. Wearable Computing: A first step toward “Personal Imaging”. *IEEE Computer,* Vol.30, No.3, 1997.

28. Muller, M. J., & Kuhn, S., Eds. Participatory design. Special issue of *Communications of the ACM,* 36 (6), 1993.

29. Nixon, P., Lacey, G. and Dobson, S. (eds.) *Proceedings of 1st International Workshop on Managing Interactions in Smart Environments (MANSE ’99),* London: Springer Verlag, 1999.

30. Norman, D. *The Invisible Computer: Why Good Products Can Fail, the Personal Computer Is So Complex, and Information Appliances Are the Solution.* Cambridge, MA: MIT Press, 1998.

31. Rekimoto, J. Time-machine Computing: A Time-centric Approach for the Information Environment. In *Proceedings of the 12th annual ACM symposium on User interface software and technology (UIST ’99),* pp. 45-54, ACM Press, 1999.

32. Rheingold, H. *Virtual Reality.* New York, NY: Simon & Schuster, 1991.

33. Robertson. G., Czerwinski, M., Larson, K., Robbins, D.C., Thiel, D., van Dantzich, M. Data Mountain: Using Spatial Memory for Document Management. In *Proceedings of the 11th annual ACM symposium on User interface software and technology (UIST ’98),* pp. 153-162, ACM Press, 1998.

34. Simon, H. *The Sciences of the Artificial.* Cambridge, MA: MIT Press, 1969.

35. Simon, H. *The Sciences of the Artificial.* 3rd ed. Cambridge, MA: MIT Press, 1996.

36. Sorensen, R. A Comparison of Software Development Methodologies. *CrossTalk,* January, 1995. URL: http://www.stsc.hill.af.mil/CrossTalk/1995/jan/Comparis.asp

37. Spence, R. and Apperley, M. *Focus on Information Technology: The Office of the Professional.* Video, Imperial College Television Studio, London, 1981.

38. Spence, R., Apperley, M., Data base navigation: an office environment for the professional. *Behavior and Information Technology,* vol. 1 no. 1, pp. 43-54, 1982.

39. Spence, R. and Apperley, M. *The Bi-focal Display.* Video, Imperial College Television Studio, London, 1983.

40. Tollmar, K., Sandor, O and Shömer, A. Supporting Social Awareness @Work, Design and Experience. In *Proceedings of CSCW 96,* pp. 298-307, ACM Press, New York, 1996.

41. Weilenmann, A. and Holmquist, L.E. Hummingbirds Go Skiing: Using Wearable Computers to Support Social Interaction. In *Pro-*

ceedings of Third IEEE International Symposium on Wearable Computing (ISWC '99), pp. 191-192, IEEE Press, 1999.

42. Weiser, M. The Computer for the 21st Century. *Scientific American,* 265(3), pp. 94-104, 1991.
43. Wellner, P., Mackay, W. and Gold, R. (Eds.) Computer-Augmented Environments: Back to the Real World. Special issue of *Communications of the ACM,* 36(7), 1993.

Flip Zooming: Focus+Context Visualization of Linearly Ordered Discrete Visual Structures

Lars Erik Holmquist

Abstract. The focus+context visualization technique *flip zooming* was developed to present data sets that can be represented as collections of linearly ordered visual elements, such as the pages of a document or a collection of images. The technique works by laying out the elements 2-dimensionally in a left-to-right, top-to-bottom fashion that reflects the linear ordering of the elements. The user move an element to the focus by clicking on it, or by moving the focus forwards or backwards to an adjacent element in the sequence. The chosen element then zooms up to a readable size, while the other elements shrink accordingly. Since the linear ordering is preserved, users have access to both a detailed view of one element and an overview of the remaining elements presented in the correct sequence. Flip zooming has been implemented in a number of prototypes, including a text-only web browser, an image browser, and a browser for hierarchically ordered image collections. During the course of the implementations, user experience motivated a move from a space-preserving layout strategy (i.e. filling the display with as much information as possible) to a place-preserving one (i.e. trying to maintain the positions of visual elements as far as possible). Currently, the most promising application area for flip zooming is for use in devices with small displays, e.g. hand-held computers.

1 Introduction

Many of the information sets that we encounter in daily life consist of a number of discrete elements which can be viewed individually, but which make more sense when placed within the context of a certain linear ordering. For instance, the pages of a book or a document can be read individually; but most of the time, one wants to read them in the correct sequence. In a calendar, each day might be viewed individually, for instance to check the current day's appointments; but many times it makes more sense to see each day's entries in the context of the preceding and following days. For a presentation, each slide accompanies a certain part of the speech; but in the flow of the presentation each slide will build on the previous and lead into the next. In each of these cases, we are not only interested in the switching from one item to another; we also want to know how far into a book we are and how much is left; if this week is more crowded with appointments than the next, and if there are any major holidays coming up that we should be aware of; if a boring speech will soon be over so that we can go home; and so on.

When we are using these objects, there are many physical clues that help us answer such questions. Books have a certain thickness, and by inspection we know approximately how far we have read and how much is left. Similarly, our paper calendars are easy to open at approximately the right place, and can be flipped through quickly to find a free spot for a meeting. And when listening to a boring presenter, we can see simply by the thickness of the stack of slides he or she is handling whether the pain will be over soon or whether it is time to think of a plausible excuse and make a quick exit. Electronic media rarely have these properties, and perhaps this is one of the reasons that many people prefer real books, paper calendars and transparencies over using computers. In particular, a computer screen is always of a limited size, which means that it can be very hard to get an overview of a material that is too large to be presented all at once on the screen.

However, some of the inherent limitations of computer screens might be overcome with novel display strategies. In our work, we have been exploring how to efficiently show large amounts of information on a limited display area, giving users visual access to both detail ("focus") and overview ("context"). This is often referred to as *focus+context visualization.* The *flip zooming* focus+context visualization technique was initially developed with the intention of displaying documents, but has proven useful to display other data, such as image collections. In the following, we give an account of related visualization techniques, followed by a discussion of different types of visual representations. The flip zooming technique is then described, followed by an account of how it has been implemented in a number of prototypes. Finally, conclusions based on our experience with the flip zooming prototypes are drawn, and future work is outlined.

2 Focus+Context Visualization

The problem of displaying large amounts of information on a limited display can be approached in a variety of ways. For instance, interactive techniques such as *dynamic queries* can be used to enable users to interactively cut down the amount of information according to some desired criteria [24]. Various intelligent filters, so-called *software agents*, have been proposed to automatically reduce the amount of information that reaches the user even before it is visualized [19], and so on.

However, if we do not want to cut down or filter out any information, we are faced with the problem of how to efficiently show a very large data set on a limited display area. Consider a large map, perhaps several meters across. If we shrink it to a size small enough to show it on a desktop screen, the user will be able to see the whole map at once, but the map will probably be too small for her to make out any detail. If we let the user see a portion of the map in actual size through a scrolling window, she can scroll to any section she wants and see that in sufficient detail, but will then have lost the important overview. A better solution might be to first present the user with an overview, and then let her zoom in on a desired portion, as for instance in the *Pad* and *Pad++* interfaces [1, 20]. However, when she views the entire map she still has no access to details, and when she zooms in to reveal details the overview will still be lost!

Therefore, it would be useful if we could present *both* an overview and a detailed view of a large material. One class of solutions to this problem are termed *overview+detail*. Here, the overview and the detailed view are not presented on the same area; instead, the user can either switch between them on the same display, much like zooming, or see them presented in different parts of the screen [7, p. 285]. In contrast, *focus+context* techniques aim to *integrate* the overview and detail in the same display area [7, pp. 307-309]. By not forcing the user to divide her attention between several different display areas, such techniques aim to provide a more effective access to visual information in a large data set. In the following, we give an brief outline of the development of focus+context visualization techniques; for a more complete view, Card et al. [7] provides a good starting point.

The first examples of focus+context visualization were non-interactive techniques for visualization of map data [14]. With the introduction of computers, it became possible to perform focus+context visualization interactively, and an early computer-based method was the *FISHEYE View* [9], later presented in slightly different form as the *Generalized Fisheye View* [10]. Here, a framework for information filtering based on the users's current point of interest was presented, so that the user for instance could see the most important information (that which was "close to the focus") in great detail, while less important information ("farther away from the focus", according to the metaphor) was only presented in outline form. The Generalized Fisheye View was initially implemented to work on text-based displays; a contemporary visualization for graphical displays was the *Bi-Focal Display* [26]. Here, distortion in the horizontal dimension was introduced to give users access to both overview and detail in a wall-sized calendar display. It is worth pointing out that the Bi-Focal display was presented as part of a larger system, where wall-sized displays, physical icons, novel input devices, etc. aided users' navigation of a large information space.

Later, the *Perspective Wall* [18] used a 3D perspective to present temporal data, achieving a similar effect to the Bi-Focal Display, and the *Document Lens* [21] developed the concept further by combining a perspective view with a magnifying-glass effect to give combined detail and overview presentation of a document. Other techniques that use various forms of distortion to display two-dimensional data displays include the *Graphical Fisheye View* [22], which allowed users to zoom in on a node in a graph or network without disturbing the spatial relationships between nodes; and the *Rubbersheet View* [23], which enabled users to apply a 2-dimensional distortion to an image, analogous to the effect of stretching a rubber sheet mounted in a rigid frame. Such techniques have also be extended to 3-dimensional displays [8]. Techniques developed specifically for visualizing hierarchies includes the *Hyperbolic Tree Browser* [16], which mapped information organized in a tree structure onto a hyperbolic surface.

3 Visual Structures in Focus+Context Visualization

When constructing a focus+context visualization for a certain data set, it is of great importance to pay attention to the way that a data set is *represented visually.* The reason is that a focus+context visualization is a form of *view transformation*, i.e. it takes a visual structure and performs a number of operations to transform the view interactively according to the user's actions [7, pp. 17, 31-32]. Another way of describing this is to say that a focus+context visualization can be seen to constitute a *second-level visualization,* or a "visualization of a visualization"; a formal account of this is given in [5]. In practice, this means that when a focus+context technique is applied to a data set, an underlying visual structure has already been selected for that data set, and that visual representation will influence whether the focus+context visualization is successful or not.

3.1 Continuous and discrete visual structures

Many focus+context techniques have been developed to visualize a 2-dimensional continuous visual presentation of some data. These visualizations simply treat the whole visual representation as a 2-dimensional image, and apply distortion to portions of the image regardless of what the image actually represents. We can thus apply techniques such as the Document Lens [21] or the Rubbersheet View [23] to *any* 2-dimensional picture, regardless of whether that picture represents a set of document pages, a map, a graph or something else entirely.

However, some focus+context techniques, for instance the Hyperbolic Tree Browser [16], can not readily be applied to continuous presentations; these techniques are specifically developed to visualize structures consisting of discrete visual elements of some sort. The difference is that these techniques transform each element in a visual structure separately and then adjust the overall presentation to accommodate these transformations, while retaining the important structural information. For instance, when zooming in on a node in a graph using the Graphical Fisheye View [22], there is no need to distort the individual nodes – these can retain their proportions. However, the placement and size of the nodes is adjusted as necessary to accommodate the increased size of the element in focus while still correctly reflecting the structure of the graph. Compare this to the same operation in the Rubbersheet View [23] – here, the surrounding information is distorted vertically and/or horizontally, since the transformation is continuous.

Thus, we can see two major types of representations where focus+context visualizations (and other view transformations for that matter) can be applied:

- *Continuous visual structures*, such as maps and images
- *Discrete visual structures*, such as trees and graphs

Continuous representations have been very popular in focus+context visualization. Leung and Apperley [17] gave an overview of such techniques, proposing a generalized “rubber-sheet” metaphor for describing them. Another general description of how to apply a focus+context view to a continuous 2-dimensional image can be found in the work on *Generalized Detail-in-Context Views* [15]. Here it is shown how a variety of continuous transformation can be applied to a 2-dimensional surface, replicating the effects of other continuous focus+context techniques. This method can also be shown to replicate “1-dimensional” techniques such as the Bi-Focal Display and the Perspective Wall. Thus, it seems that all the multitude of continuous focus+context techniques can be represented as a continuous function applied to a 2-dimensional surface.

Among discrete focus+context techniques, there are many notable visualizations of graphs (e.g. The Graphical Fisheye View applied to graphs) and hierarchies (e.g. The Hyperbolic Tree Browser). Although graph visualizations usually aim to preserve certain 2-dimensional relations, they can take greater liberties in moving the discrete elements, than can the continuous techniques. In visualizing trees, no relations need to be preserved except those which indicate the relations between parent and child nodes, thus allowing for even greater flexibility when designing the resulting focus+context layout. The strength of the Hyperbolic Tree Browser, for instance, is that when it maps a tree structure onto a hyperbolic surface the results may not look like any traditional tree, yet the tree structure is still immediately and intuitively recognizable. Mapping a graph or a continuous 2-dimensional image onto a hyperbolic surface in the same way would probably give a much less satisfying result. On the other hand, the Graphical Fisheye View visually preserves the relationships in a graph to much greater degree, but cannot present a tree nearly as effectively as the Hyperbolic Tree Browser.

3.2 Structuring of visual elements

Most of the time the information in a discrete visual representation is not simply a collection of elements (i.e. a set), but it is *structured* in some way, which in turn will influence the structure of the visual representation. The way in which a visual representation is structured will effect a focus+context visualization in how it restricts the *traversal* of the elements, i.e. how a user can choose a new focus by moving the focus from the current element to one that is adjacent to it in the visual structure. For instance, when viewing a linear sequence of elements, say like the pages of a document, the user should be able to move back and forth in the document to preceding and following pages. On the other hand, when viewing a hierarchical structure, i.e. a tree, the user should be able to traverse the tree according to its structure of branches and leaves, and so on. In other words, the user should always be able to traverse the data set in a way which reflects its inherent structure.

It thus seems that we need to make a difference not only between discrete and continuous representations, but also in the various ways in which discrete representations are structured. Shneiderman [25] listed seven types of *data* structures: 1-dimensional; 2-dimensional; 3-dimensional; Temporal; Multi-dimensional; Tree; and Network. However, a data structure is not the same as a visual structure – in many cases there has to be a mapping from the data structure to a corresponding visual structure that can be presented on a 2-dimensional display. Apart from unstructured collections, i.e. sets, we can see three such visual structures:

- *Linear ordering,* where each element has a place in a linear (1-dimensional) sequence. Here, the user can only go to the following or the preceding element in the sequence.
- *Hierarchical ordering,* where each element has a certain place in a tree structure. This can be traversed by going up and down in the hierarchical structure.
- *Graphs,* where each element can be connected to one or more other elements to an arbitrary level of complexity. Here, the user has the most freedom since she can go to any node in any direction that there is a direct connection from the current node

Although the linearly ordered structures offer the most limited navigation, they are quite important, since they can be used to represent a wide variety of data types, and we will concentrate on this aspect for the remainder of the paper.

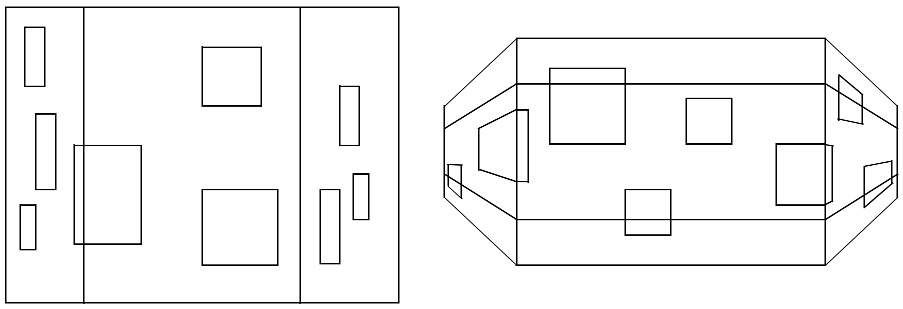


Fig. 1. Schematic diagrams of the view transformations taking place in The Bi-Focal Display (left) and The Perspective Wall (right).

3.3 Linearly ordered discrete visual structures

We can now see that there is something in common between the physical information representations that were discussed in the introduction – documents, calendars, presentation slides, etc. They consist of a number of separate visual elements, each which can be viewed on its own; but they also possess an inherent linear ordering, which must be preserved if the data set is to make sense as a whole. In other words, they are all *linearly ordered discrete visual structures*. This type of visual structure is quite common in the real world, but difficult to effectively represent on a computer screen. The most common way to do this is to use a scrollbar, which lets the user move a small window over an area that is larger than the display. However, traditional scrollbars have no means of providing an overview of the material, apart from telling the user approximately at which position the window is relative to the larger area.

Some focus+context methods have been developed for displaying an area that is wider than the available screen space; two of these are shown schematically in **Fig. 1**. The Bi-Focal display (left) [26] solved this by compressing the horizontal portion of the material left and right of the center, whereas the Perspective Wall (right) [18] used graphical distortion according to a 3-dimensional metaphor, by showing the context material as if it was receding into the background of the image on both sides of the focus. Although these methods are designed for continuous visual representations, they might also be modified to display discrete visual structures. For instance, if displaying a set of document pages, we might allow only one page in focus at the time, taking up the full focus area, and have all the preceding and following pages displayed in the left and right context areas. Navigation could then be performed in discrete steps, by moving one page forwards or backwards.

But it should also be noted that although the methods mentioned above would preserve the linear ordering of a discrete visual structure well, they are not optimal in all respects. The Bi-Focal Display introduces a heavy horizontal compression on the context areas, while keeping the vertical dimensions unchanged; this "stretching", while space-filling, gives context elements a very distorted look. The 3D-metaphor used in the Perspective Wall gives a more natural appearance to the context, since it resembles how far-away objects would look in the real world, but it is not very space efficient since a large portion of the vertical space goes unused. Also, although fitting to the perspective metaphor, the distortion introduced in the context display may make elements far from the focus hard to recognize.

4 The Flip Zooming Focus+Context Technique

With flip zooming, we initially set out to design a focus+context visualization technique for documents, in particular documents consisting of several discrete pages such as the pages of a book. However, having arrived at such a technique, we of course hoped to be able to generalize it to other types of data. We wanted our new method to share some fundamental properties with previous focus+context techniques, including:

- *Overview of the whole data set.* The entire data set should be presented simultaneously, to give the user an overview.
- *One selectable focus.* It should be possible to focus on one element, i.e. present it large enough to be readable.
- *Random access to any visual element.* In the visually presented data, the user should have instant access to the whole data set, so that any element can be moved to the focus (e.g. by point-and-click)
- *Space efficiency.* The available space should be used to display as much information as possible.
- *Preservation of linear ordering.* The inherent linear ordering of the elements should be preserved and presented in the resulting display.

Some additional goals that we wanted to achieve, that were not fully met by previous methods, were:

- *Low computational demands.* To facilitate real-time interaction, the methods should require as little calculations as possible.
- *No distortion of the text.* All continuous focus+context methods introduce some degree of distortion, which we found undesirable, especially for viewing text. We felt that a method which preserved the proportions of the visual elements would be preferable.
- *Linear traversal of visual elements.* Since the most common way to traverse a document (or, more generally, to a set of linearly ordered visual elements) is to go from one page to the next, the new method should support this by allowing the user to move "backwards" and "forwards" through the elements (e.g. by using some keyboard command)

4.1 Design process

With the above goals in mind, we started sketching a variety of different possibilities. The basic idea was to lay out a document as separate pages and then let the user zoom in on a particular page. Some obvious ways

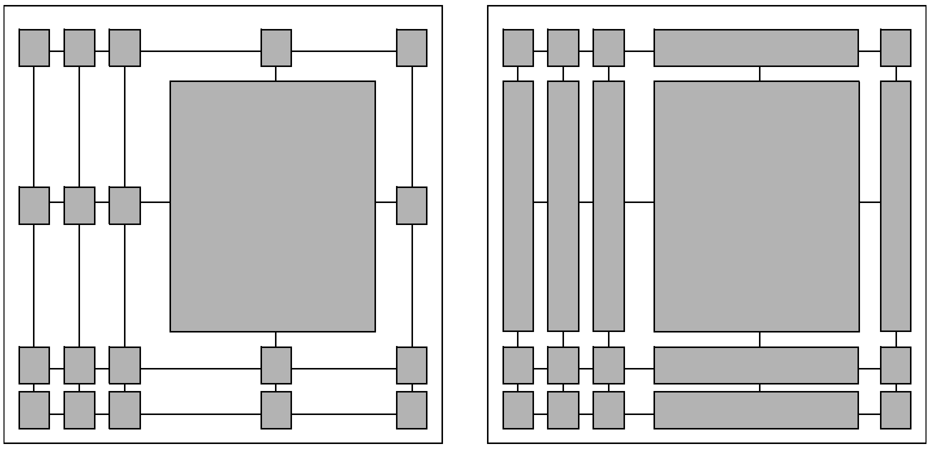


Fig. 2. Some concepts sketches; the gray boxes represent document pages. Left, proportions are preserved; right, space is used more efficiently by distorting the context elements.

of achieving this are sketched in **Fig. 2.** In the left figure, which is similar to the Graphical Fisheye View [22], when a user zooms in on a page, the other pages shrink accordingly to accommodate the current focus; their proportions are preserved. This has the advantage of not distorting the content of the pages, thus making them easier to recognize, but it does not maximize the use of screen space, since there are large areas which are not used to display information. In the right variant, which is similar to the Rubbersheet View [23], certain pages are expanded vertically or horizontally to the same degree as the focus. This has the advantage of using the available space more efficiently to display information, but it also introduces visual distortion to the content of the pages.

It seemed that there was no way to achieve a display which preserved the linear ordering while using the space efficiently and without introducing distortion to the individual elements. However, after more sketching, we realized that we were placing a too hard restriction on our display: To preserve linear, i.e. *1-dimensional* ordering, we were in fact preserving a *2-dimensional* ordering. We realized that if we found some other way to preserve the linear ordering, we might break the rigid frame that had constricted the previous variants, thus utilizing the full display area more effectively. The resulting technique, which allowed users to “flip through” a data set, and “zoom in” on the element which most interested them without losing the overview, was termed *flip zooming*.

4.2 Flip Zooming

The easiest way to explain how flip zooming works is to refer to **Fig. 3.** Here, it is schematically shown how a set of linearly ordered discrete visual elements are first laid out on a 2-dimensional surface (left). This could for instance be thumbnails (i.e. miniature graphical representations) of the pages in a document. The figure then shows how the same data collection looks when the user has chosen to focus on a certain element; the element is zoomed up to a readable size, while the surrounding elements are shrunk accordingly to accommodate the focus (right). Compare the view in this figure with the corresponding views in **Fig. 2**. It can be noted that flip zooming allows for the context elements to be of a much larger size (approximately four times the size in these examples), thus preserving more readability, while still not introducing any distortion to the individual elements.

However, the linear ordering of the document pages or other data must be preserved in some way. In focus+context methods which completely preserve 2-dimensional relationships, this is done automatically. In flip zooming, it is instead done through the convention of always laying out the elements in a *left-to-right, top-to-bottom* fashion. **Fig. 4** illustrates how this works. When the view is not zoomed, the ordering of the elements is quite straight-forward. When the user zooms in, the view changes, but the ordering is preserved according to the left-to-right, top-

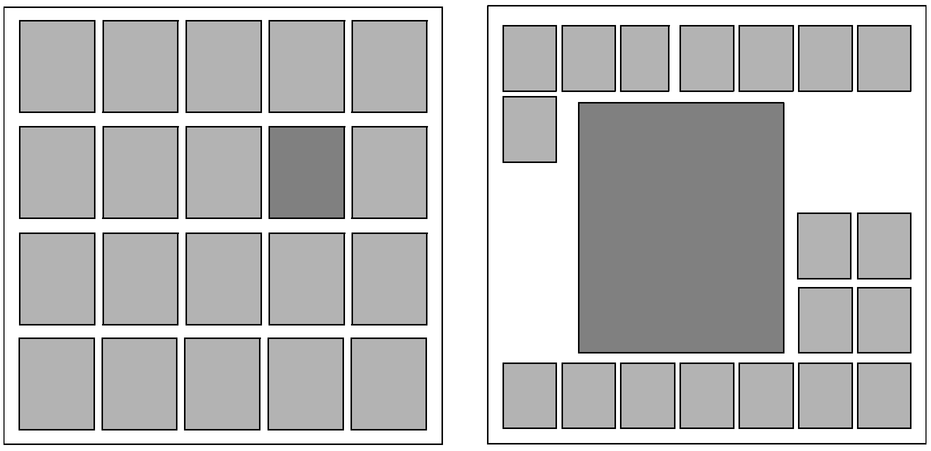


Fig. 3. Flip zooming view of 20 linearly ordered elements. Left, un-zoomed view; right, zoomed view.

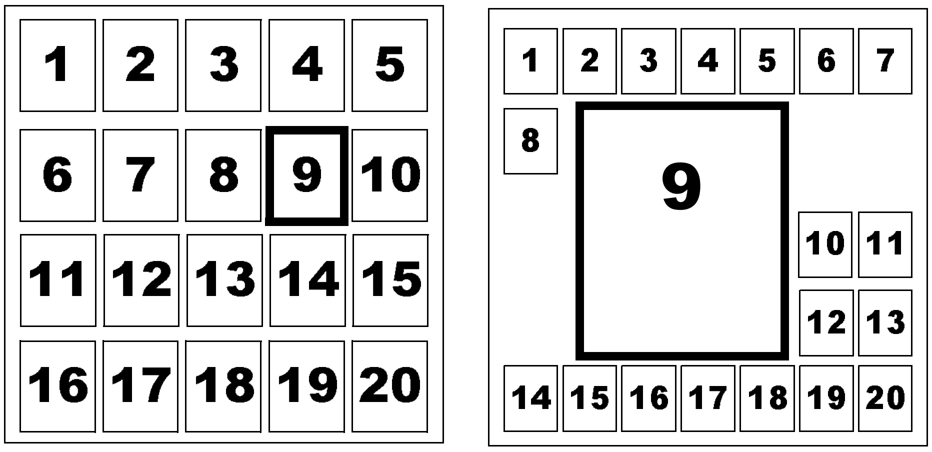


Fig. 4. A view of the left-to-right, top-to-bottom ordering.

to-bottom scheme. Every element which is *before* the focus in sequence, can be found *to the left of and / or above* the focus; every element that comes *after* the focus can be found *to the right of and / or below* the focus.

It should be noted that the exact results of this layout scheme is not strictly defined. In particular, there is quite a lot of freedom in deciding exactly how to construct a zoomed layout, both when it comes to the placement of the focus and the layout of the context elements. This means that it might be hard to tell where an element might appear when focused on, since this might not be the same as its position when not in focus. **Fig. 3** provides a good illustration of this, since the zoomed element in the right figure does not seem to have appeared very close to its previous position in the left figure (compare this to **Fig. 2**, where the element is more or less fixed in the same place when zooming).

We can now see how flip zooming fulfills the criteria we initially set up:

- *Overview, focus, and random access.* All these fundamental focus+context properties are fulfilled by flip zooming; the whole data set is presented visually, and any element can be moved into focus (i.e. zoomed up to a readable size) by clicking on it.
- *Space efficiency.* Because of the mapping to two dimensions, flip zooming takes advantage of the shape of most displays, by efficiently using the vertical as well as the horizontal screen space for context

information. Also, compared to methods which use two dimensions but preserve a strict 2-dimensional ordering, flip zooming uses space more efficiently with the result that the context elements can be made larger; compare **Fig. 2** (left image) with **Fig. 3** for an illustration of this.

- *Preservation of linear ordering.* With the consistent left-to-right, top-to-bottom ordering, flip zooming preserves the linear ordering of the elements, although it maps it to two dimensions and modifies it as necessary to make the most effective use of the display.
- *Low computational demands.* Flip zooming requires very little computational power. The visual elements will need to be scaled to the right size, but once this has been done, the only computation that needs to take place is that required for calculating the placement of the visual elements. Since the visual elements can be cached once they have been calculated, flip zooming systems can be very responsive even on machines with very limited processing power.
- *No distortion of individual elements.* Flip zooming does not impose any visual distortion on the context material. Instead, all elements keep their proportions and are merely reduced in size, which means that their visual properties are better preserved. Flip zooming does not introduce vertical or horizontal distortion to context elements; compare **Fig. 2** (left image) with **Fig. 3** for an illustration of this.
- *Linear traversal of visual elements.* A user can always move to the next or previous element in the sequence on the display. (Exactly how this is done is implementation dependent but it could be done for instance by using the “left” and “right” arrow keys, or by providing some GUI buttons.)

However, there are also some obvious disadvantages with the method:

- *The placement of focus and context elements is not fixed.* Whenever arranging a flip zooming layout with one element in focus, there may be several (equally correct) ways in which the elements can be placed. This might confuse users since it can be hard to predict the result of a certain action, e.g. where a new focus will appear when focusing on a certain item.
- *Zooming changes the row and column layout.* When the user zooms in on an element, it is necessary to change the number of rows and columns to accommodate the focus; this may make the transition between different views confusing.

- *Limit to how much data can be displayed.* At a certain point, when there are too many elements visible, the context elements will be too small to be useful (the theoretical limit is context elements that are just one pixel in size, but in practice this happens much earlier). To combat this, some strategy for displaying more complex structures, e.g. hierarchies, might be needed.

5 Flip zooming prototypes

To experiment with and evaluate the technique, we have implemented flip zooming in several applications. In the following we will describe three prototypes:

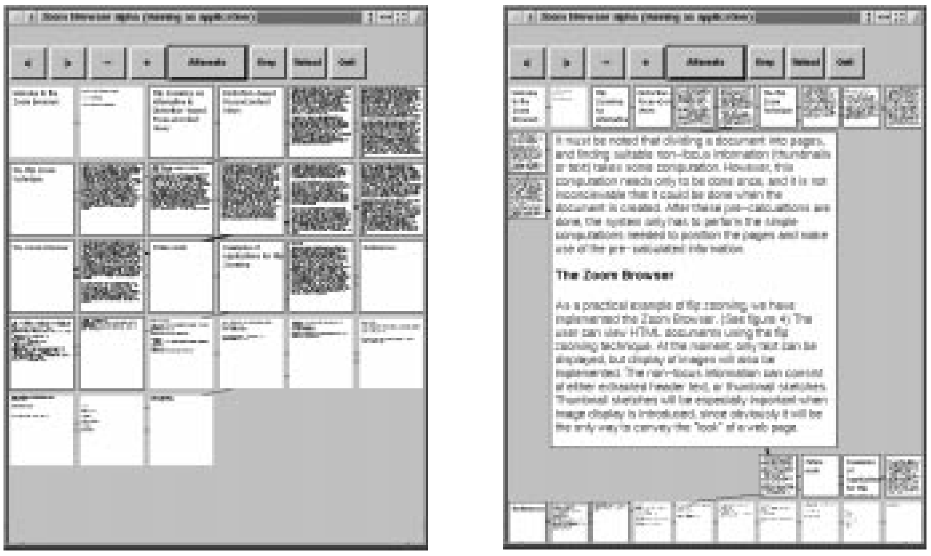
- *The Zoom Browser,* a text-only web browser
- *The Flip Zooming Image Browser,* an application for browsing images
- *The Hierarchical Image Browser,* an application that allowed browsing of more complex image collections

When describing the prototypes, we will take special note of these points:

- *Layout*, i.e. the various ways in which the prototypes arrange the individual visual elements.
- *Lessons from user experience,* i.e. what we learned from evaluating the prototypes.

5.1 The Zoom Browser

The first implementation of flip zooming was the *Zoom Browser* [11, 12]. It was a text-only browser for the World Wide Web and acted as a test-bed for exploring the feasibility of flip zooming. The Zoom Browser presented a view of one or more web pages by dividing the text into a number of separate chunks, and then presenting the chunks as pages of text on a flip zooming display. The Zoom Browser was developed using an early version of the language Java. This meant working under some limitations, most notably that the Java language at that time was quite slow; for this reason, it was fortunate that flip zooming has



It must be noted that dividing a document into pages, and finding suitable non-focus information (thumbnails or text) takes some computation. However, this computation needs only to be done once, and it is not inconceivable that it could be done when the document is created. After these pre-calculations are done, the system only has to perform the simple computations needed to position the pages and make use of the pre-calculated information.

The Zoom Browser

As a practical example of flip zooming, we have implemented the Zoom Browser. (See figure 4) The user can view HTML documents using the flip zooming technique. At the moment, only text can be displayed, but display of images will also be implemented. The non-focus information can consist of either extracted header text, or thumbnail sketches. Thumbnail sketches will be especially important when image display is introduced, since obviously it will be the only way to convey the "look" of a web page.

Fig. 5. The Zoom Browser; starting view (left) and zoomed view (right).

very low computational demands (as pointed out above). Also, the lack of any readily-available HTML parsing and rendering code meant that we had to limit the display of web pages to a text-only format. These disadvantages were however more than compensated for by the advantages of using Java, including easy implementation of web services and the possibility of making the prototype instantly available as a demo on the web.

Layout. The layout used in the Zoom Browser was designed to be *space-preserving* rather than *place-preserving.* To maximize the usage of screen space and leave as little area of the screen unused as possible, it was necessary to allow focus and context elements appear at different positions depending on what element was in focus. Furthermore, the layout allowed for some freedom in the placement of the focus element. This was used to place the current focus element in a position that could be assumed to be close to the user's current focus of attention. Depending on whether the user had clicked on a context element, or used a command shortcut to advance to the next element, the focus element was placed as close as possible to the selected context element, or the previous placement of the focus element, respectively. However, this also meant that the same focus element could appear at different positions depending on which action had been performed to arrive at the focus.

Lessons from user experience. The Zoom Browser was not subjected to any formal user testing. However, we let several users experiment with the prototype and comment on its features, and received some valuable feedback, including:

- *The layout was sometimes confusing.* Our decision to make the layout scheme space-preserving rather than place-preserving unfortunately resulted in situations where it was hard for users to predict where a certain element would appear during interaction. On the other hand users also complained that the browser did not seem to be using the display space as effectively as possible (even though it can be shown that it actually did use the space as effectively as possible most of the time). This was an indication that there is an important balance to be struck between making a display that is too “crowded” and confusing, and between making a clear and understandable display, which might not use the available screen-space as effectively as possible.
- *Users liked the quick access to detail and overview that the Zoom Browser provided.* Several users immediately expressed sentiments like “I want this!” or “When can I have this for my word processor?”. These user reactions showed that for viewing text, flip zooming seemed to be a very compelling technique.

5.2 The Flip Zooming Image Browser

After the initial experiences with the Zoom Browser, we wanted to explore how we might use the flip zooming technique on other kinds of data sets. Browsing image collections seemed a natural progression from browsing text pages, since they too are comprised of individual visual elements. Many image collections do not have a strong inherent linear ordering, but can be accessed in any order, something that might make them a less obvious subject for flip zooming. On the other hand, the overview and easy access provided by flip zooming might make image browsing a suitable choice. Thus we decided to implement a *Flip Zooming Image Browser.*

We implemented the Flip Zooming Image Browser in Java, like previously the Zoom Browser. At this time, the Java language was still too limited to smoothly scale images in real-time; instead we decided to use a set of pre-scaled thumbnails of a fixed size that were stored along with the full-size images. This limited the flexibility of the application somewhat in that it was difficult to adapt the layout to different screen and window sizes. It was however evident that the advantages of using Java, such as the short development time and the possibility of making material available directly on the web, again outweighed the shortcomings. Fortunately, the continued development of the Java language meant that the problems could be overcome in the next prototype (below), making this original Image Browser a stepping stone that was most valuable in the user experience it provided.

Layout. The Image Browser, like the Zoom Browser, used a space-preserving rather than place-preserving layout strategy. However, it was slightly less effective than the Zoom Browser, since it could not scale images in real-time to make the most of the available display area.

Lessons from user experience. We performed a formative evaluation of the Flip Zooming Image Browser [3]. The study emphasized qualitative results and was an effort to find possible areas of improvement of the prototype as well as of flip zooming in general. In the study, ten users performed a variety of tasks while using the Flip Zooming Image

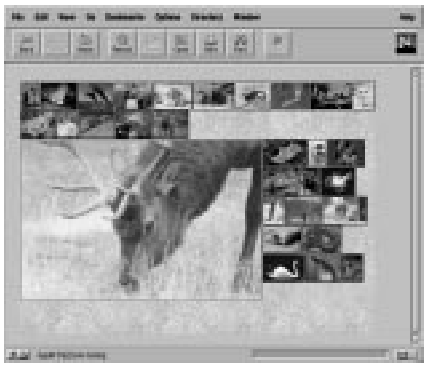
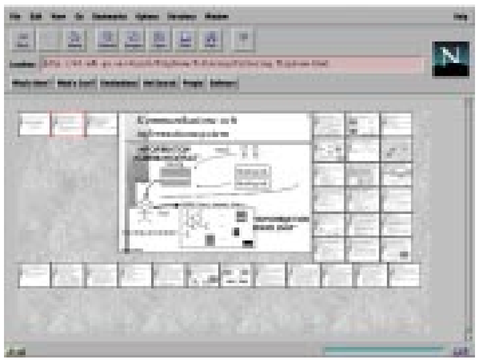



Fig. 6. The Image Browse running as a web-page applet; browsing images (left) and PowerPoint slides (right).

Browser to browse two collections of images: one consisting of pictures of different animals, and one consisting of a collection of PowerPoint slides with a mixture of text and graphics. The tasks were designed so that they required the use of different search strategies, such as image recall, text search, etc. The tasks were for instance: “find a slide with a certain word”, “find out if there are more pictures of one type of animal than another”, etc.

To provide a contrasting experience, the subjects also performed similar tasks using the Adobe Acrobat PDF Reader’s overview+detail display to browse PDF versions of the images, and a traditional web browser to browse web pages generated from the presentation. However, the applications were too different for them to be directly comparable (for instance, the Acrobat Writer did not provide thumbnails, only blank rectangles, making for a severely limited overview). After each session, users were subjected to semi-structured interviews, the results of which were collected in various categories (e.g. “Good overview”, etc.). Some points which were mentioned by at least 50% of the subjects were:

- *Overview.* 10 (i.e. all) users said that the Flip Zooming Image Browser provided a good overview of the material; in comparison, 3 said that the web page provided a good overview and only 1 that the Acrobat Reader provided a good overview. Thus, we could conclude that users perceived the flip zooming as a useful approach to giving an overview of a large material.
- *Clear or confusing display.* 5 users specifically pointed out that they liked the clear layout of the Adobe Acrobat Reader (one user called it “neat and tidy”). In contrast, no users said that the Image Browser had a clear display, but 5 users specifically pointed out that they felt the Image Browser’s display was confusing. This was likely a reflection of the trade-off between space- vs. place-preserving layout strategies, and we concluded that even though we managed to use the space effectively this was not particularly appreciated by users (no user mentioned it). Instead, we felt it might be fruitful to explore methods to make the display less confusing by keeping elements fixed in the same place, even if that meant a trade-off in screen-space usage.
- *No / too small thumbnails.* 5 users thought that the thumbnails displayed by the Image Browser were too small (despite our space-preserving strategy). This might be because the size of the thumbnails

was fixed, due to the technical limitations mentioned above. 9 users complained that the Acrobat Reader showed no thumbnails, only outlines, indicating (not surprisingly) that thumbnails are considered useful when browsing image collections, and should be included if possible.

Suggestions from users included to allow for a full zoom-in on any picture on the display, and to separate the thumbnails from the focus image in a manner similar to the Acrobat Reader's display. We also noticed that the overview provided by flip zooming allowed users to initially give approximate (and fairly, but not 100%, correct) answers to some tasks, which they then would rapidly refine through a more extensive search of the material. Thus we saw that flip zooming could be used by novice users to first get an overview or a general "feel" for a visual material, and then allowed for a more detailed exploration.

5.3 The Hierarchical Image Browser

Based on the experience gained from the Image Browser, we constructed a new prototype: The Hierarchical Image Browser [2, 13]. Thanks to the improvement of the Java language, it was now possible to scale images in real-time, allowing for a much more flexible display. Furthermore, we wanted to explore how we might visualize more complex collections of images, in particular those which were ordered according to certain categories; thus the introduction of hierarchies. By using the fact that Java allows for nesting of interface components, hierarchical displays was achieved quite easily. The flip zooming application was generalized to allow the display of not just images, but any kind of interface components; and since the flip zooming display in itself is an interface component, nesting followed automatically. The display displays another flip zooming display as if it were just another visual element, thus allowing for introduction of any number of nested flip zooming views.

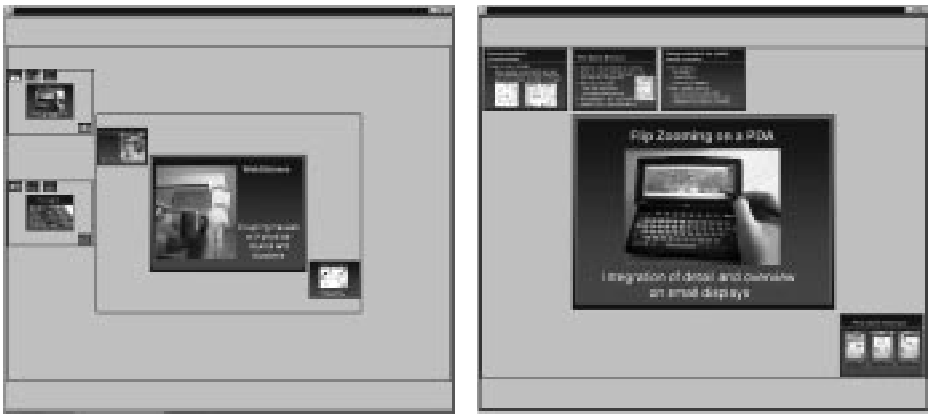


Fig. 7. The Hierarchical Image Browser; showing a full overview of a structured PowerPoint presentation (left) and zoomed in on one section (right). A full-screen view, where a single slide fills the whole display, is also available but not shown here.

In practice, the user would provide the application with a file structure of one or more directories containing images, where each directory in turn can contain sub-directories and so on. The application creates a main flip zooming display for the whole directory tree, then creates a new flip zooming display inside the main display for each individual directory, and recursively creates new nested displays for each subdirectory, and so on. Thus the entire directory tree is represented as flip zooming components, which in turn may contain images and/or further sub-components.

Layout. The layout of the Hierarchical Image browser was heavily influenced by the formative evaluation of the first Flip Zooming Image Browser. In this evaluation we found that users were confused by the fact that the focus image could appear on different places in a not very predictable manner. This was because of our effort to create a space-efficient display.

For this prototype, we decided to instead implement a place-preserving layout, and keep the focus element in a fixed location at the middle of the display. This would not place the image at the same location as the corresponding thumbnail when it was clicked on (as in the Zoom Browser), but it would make it appear consistently on the same place every time, making for a more predictable interaction. These changes, however, meant that we had to make the layout much less space effective: to maintain the linear ordering, we had to leave some portions of the display area unused, depending on which image was in focus. But based on the evaluation of the previous prototype, we felt that this was a necessary step to take.

Additionally, the Hierarchical Image Browser allowed the user to control the amount of context displayed, by enabling the user to zoom in completely on one branch of the directory tree if needed. When doing so, only the images in the current directory were shown, not the outside context. Thus, users could perform a form of visual "pruning" of the hierarchy, so that context that was not relevant to the current task was not displayed.

Lessons from user experience. To evaluate the Hierarchical Image Browser, we decided to perform an "evaluation by real usage", choosing some real expert users as subjects – namely, ourselves. We therefore made sure that the prototype was sufficiently robust to use it as a presentation tool, acting as a direct replacement to Microsoft PowerPoint. Presentations slides still had to be prepared in PowerPoint, but by saving them as JPG images, they could be viewed directly in our own prototype.

We used the Hierarchical Image Browser for about six months in a variety of situations, including formal talks at conferences (e.g. at SIGGRAPH '98) and at open walk-in demonstrations (e.g. at CSCW '98). At the demonstration, we ran a flip zooming display of a presentation on a large touchscreen in conjunction to the demonstration, to be able to quickly provide illustrations to certain points in the demonstration. We also used the prototype at external presentations of our work at other research labs, when giving presentations for local industrial partners, during lectures at the local university, and so on.

During this usage, we found the Hierarchical Image Browser to be a good help in presenting structured presentations, especially when several different topics where to be covered using different sets of slides. These slides were then simply collected in different directories, and presented one by one by zooming in on each slide set. Especially useful was the ability to be able to quickly jump to some slide (for instance when an audience member asked a question) by zooming out to get an overview of the presentations, find the slide by visual inspection of the whole set, and then zoom in again to display the slide in full size.

6 Discussion

With flip zooming, we have introduced a focus+context visualization technique which preserves the linear ordering of a set of discrete visual elements, while making efficient use of a 2-dimensional display area. This is done by mapping the 1-dimensional ordering to two dimensions, thus using both vertical and horizontal space efficiently. However, this approach gives rise to some problems. Most previous focus+context techniques, in particular continuous methods but also those that deal with discrete visual representation, are strictly *place-preserving,* i.e. they keep the spatial relationships between visual elements intact when the user zooms in. Flip zooming does not require this; when the user zooms in on an element, it may change its position to a completely different part of the display, the context elements may also move, and the number of rows and columns may change. All this can of course serve to confuse users when interacting with a flip zooming system.

The reason for the changing display, however, was to make flip zooming *space-preserving,* i.e. to use the available display space as efficiently as possible. Many focus+context techniques do this, usually by the introduction of spatial distortion (vertical, horizontal, or resembling a 3D-dimensional effect). Flip zooming does not introduce spatial distortion of the individual elements, but this is done at a price: to keep the technique as space-preserving as possible, we needed to loosen the place-preserving restrictions, with the problems outlined above as a result.

That the unpredictable layout indeed was a serious problem was clearly born out by the user experience, and in the last prototype, we attempted to address this. We decided that it was worth trading some of the available display area for a more easily-understood layout. To accomplish this, we changed the layout strategy from being primarily space-preserving in the first two prototypes to a more place-preserving approach in the last. In particular, this meant keeping the position of the focus element at the centre of the display, rather than placing it at the position that would permit us to use the available space most efficiently.

The result was a layout that gave a predictable response during user interaction, albeit at the cost of being less information dense.

We believe that this result, where we saw the necessity to trade a certain amount of information density for a clear and predictable display, has implications for focus+context visualization in general. It is not enough to provide a totally space-efficient display if users are confused during interaction. At the same time, it is not always necessary to maintain a totally strict space-preserving layout (or, as in for instance the case of document visualization, to strictly preserve a 2-dimensional layout for a linearly ordered data set). Even when accounting for the introduction of a more place-preserving layout the flip zooming technique is still quite space efficient when compared to many other techniques, and through the course of the prototype development it has achieved a promising middle-ground between space-efficiency and user friendliness

7 Future work

With flip zooming, we have so far only started to examine the implications of the trade-offs of space-preserving vs. place-preserving layouts. We see an opportunity for much user-testing of both current and forthcoming prototypes, hopefully to be able to construct guidelines for when and to how high a degree such a trade-off should be considered. Also, real benchmarking of flip zooming versus other focus+context techniques is yet to be done, and there is a need for both quantitative and qualitative comparative studies in this area. In addition to this, we think that flip zooming can be generalized to other usage situations and data types than those described in this paper.

Currently, one of the most promising areas for flip zooming small displays, such as those found on hand-held computers. The fact that flip zooming has low computational demands makes it extra suitable for such applications. We have so far developed two prototypes that used flip zooming to display data on small displays: *WEST* [6], a web browser for small terminals, and *PowerView* [4], an integrated activity calendar which allowed users to retrieve personal data commonly stored on a PDA, such as contacts, meetings, to-do-items, e-mails, etc. In both these applications, the limited display area made it necessary to introduce various strategies to filter and hide data, such as hidden hierarchical structures in WEST and usage-sensitive context displays in PowerView. This is indicative of that focus+context visualizations such as flip zooming are in themselves not sufficient when presenting data on small displays; instead, there is need for combinations where various data selection techniques are combined with visualization in an intelligent manner.

Thus, it should be fruitful to explore of how focus+context visualization and usage context (in a wide sense) can be utilized on handheld computers. If a PDA is combined with a mobile phone, we will have access to information such as who he user is currently talking to on the phone. This information can be used to adapt the context display so that for instance when a certain person calls, only the information relevant to this specific caller is displayed on the PDA. Other types of contextual information might also be used to adapt the information display on a mobile device. The size of the focus and thumbnails might change according to the light conditions; the level of ambient noise might affect if audio cues might be used to support interaction, and so on. Thus, the concept of “context” in focus+context visualization can take on a whole new meaning, and flip zooming, being a light-weight, adaptable technique, should adapt well to this change.

8 Acknowledgements

Thanks to all my collaborators in the various projects on which this paper is based, in particular Staffan Björk and Johan Redström, who contributed many important developments, and Christopher Ahlberg, who was my supervisor in the Zoom Browser project. The Flip Zooming Image Browser was implemented by Roberto Busso. This work was funded by NUTEK through the IVES project; by KFB through the Internet Project; and by The Foundation for Strategic Research through the Interactive Institute. Additional funding was received from SITI through the Mobile Informatics research program.

9 References

1. Bederson, B., Hollan, J., Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. *Proceedings of ACM UIST ’94,* ACM Press, 1994.
2. Björk, S. Hierarchical Flip Zooming: Enabling Parallel Explorations of Hierarchical Visualizations. *Proceedings of Advanced Visual Interfaces 2000,* ACM Press, 2000.
3. Björk, S. and Holmquist, L.E. Formative Evaluation of a Focus+Context Visualization Technique. *Proceedings of HCI ’98* (poster), The British HCI Society, Sheffield, UK, 1998.
4. Björk, S., Holmquist, L.E., Ljungstrand, P. and Redström, J. PowerView: Structured Access to Integrated Information on Small Screens. *Extended Abstracts of CHI 2000,* ACM Press, 2000.
5. Björk, S., Holmquist, L.E. and Redström, J. A Framework for Focus+Context Visualization. *Proceedings of IEEE Information Visualization ’99,* IEEE Press, 1999.
6. Björk, S., Holmquist, L.E., Redström, J., Bretan, I., Danielsson, R., Karlgren, J. and Franzén, K. WEST: A Web Browser for Small Terminals. *Proceedings of ACM UIST ’99,* ACM Press, 1999.
7. Card, S.K., Mackinlay, J.D and Shneiderman, B. (eds.) *Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers, San Francisco, California, 1999.
8. Carpendale, M.S.T., Coperthwaite, D.J. and Fracchia, F.D. Extending Distortion Viewing from 2D to 3D. *IEEE Computer Graphics and Applications,* July August, 1997.
9. Furnas, G.W. The FISHEYE View: A New Look at Structured Files. Bellcore Technical Report, 1981. Reprinted in: Card, S.K., Mackinlay, J.D and Shneiderman, B. *Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers, Inc., San Francisco, California, 1999.
10. Furnas, G.W. Generalized Fisheye Views. *Proceedings of CHI '86,* pp. 16-23, ACM Press, 1986.
11. Holmquist, L.E. Focus+Context Visualization with Flip Zooming and the Zoom Browser. *Extended Abstracts of CHI ’97,* ACM Press, 1997.
12. Holmquist, L.E. The Zoom Browser: Showing Simultaneous Detail and Overview in Large Documents. *Human IT,* vol. 2 no. 3, ITH, Borås, Sweden, 1998.
13. Holmquist, L.E. and Björk, S. A Hierarchical Focus + Context Method for Image Browsing. *SIGGRAPH '98 Sketches and Applications,* ACM Press, 1998.

14. Kadmon, N., and Shlomi, E. A polyfocal projection for statistical surfaces. *Cartograph,* J. 15, 1, 36-40, 1978.
15. Keahey, T. The Generalized Detail-In-Context Problem. *Proceedings of IEEE Visualization ’98, Information Visualization Symposium,* IEEE Press, 1998.
16. Lamping, J., Rao, R., Pirolli, P., A focus+context technique based on hyperbolic geometry for viewing large hierarchies. *Proceedings of CHI '95,* ACM Press, 1995.
17. Leung, Y.K, Apperley, M.D, A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Transactions on Computer-Human Interaction,* vol. 1 no 2, pp. 126-160, 1994.
18. Mackinlay, J. D., Robertson, G. G., Card, S. K, The Perspective Wall: Detail and Context Smoothly Integrated. *Proceedings of CHI '91,* pp. 173-179, ACM Press, 1991.
19. Maes, P. Agents that Reduce Work and Information Overload. *Communications of the ACM,* Vol. 37, No.7, pp. 31-40, 146, ACM Press, July 1994.
20. Perlin, K. and Fox, D. Pad: An Alternative Approach to the Computer Interface. *Proceedings of SIGGRAPH ’93,* ACM Press, 1993.
21. Robertson, G.G., Mackinlay, J.D., The Document Lens. *Proceedings of UIST '93,* pp. 101-108, ACM Press, 1993.
22. Sarkar, M. and Brown, M.H. Graphical Fisheye Views of Graphs. *Proceedings of CHI ’92,* pp. 83-91, ACM Press, 1992.
23. Sarkar M., Snibbe, S.S., Tversky, O.J. and Reiss, S.P., Stretching the Rubber Sheet: A Metaphor for Viewing Large Layouts on Small Screens. *Proceedings of ACM UIST '93,* pp. 81-91, ACM Press, 1993.
24. Shneiderman, B. Dynamic Queries for Visual Information Seeking. *IEEE Software,* 11(6), pp. 70-77, 1994.
25. Shneiderman, B. The eyes have it: A task by data type taxonomy for information visualizations. *Proceedings of IEEE Visual Languages ’96,* pp. 336-343, IEEE Press, 1996.
26. Spence, R., Apperley, M., Data base navigation: an office environment for the professional. *Behavior and Information Technology,* vol. 1 no. 1, pp. 43-54, 1982.

A Framework for Focus+Context Visualization

Staffan Björk, Lars Erik Holmquist and Johan Redström

Abstract. Focus+context visualization techniques aim to give users integrated visual access to both details and overview of a data set. This paper gives a systematic account of such visualization techniques. We introduce the notion that there are different levels of information visualization, with focus+context being a second-level visualization, and illustrate this with examples. We then provide a formal framework for describing and constructing focus+context visualization and relate this to the examples. A description of a software framework based on the principles of the theoretical framework follows, and we give some examples of how different focus+context visualization applications have been constructed using this framework. Finally, we discuss the implications of the formal framework and outline some future work in this area.

Keywords. Focus+context visualization, information visualization, fisheye views, formal methods, theory

1 Introduction

Information visualization is widely acknowledged as a powerful way of helping users make sense of complicated data, and a great number of methods for visualizing and working with various types of information have been presented. However, all information visualization techniques will have to comply to one inherent limitation: they will need to limit themselves to the available area of a computer screen. A common solution to this problem is to provide some kind of movable view-port to the data, which can be controlled through the manipulation of scrollbars or other means. Zooming interfaces have also been introduced to let users control the amount of data shown, e.g. [3]. Sometimes, however, it might be important to give users access to both overview and detailed information at the same time; such techniques include [21], with separate areas for overview and detail-on-demand information.

Here, we will concentrate on a certain family of techniques, that attempt to integrate both detail and overview on the same display area in an effort to not divide the user's attention. Some terms which have been used for such techniques include *fisheye views, distortion-based presentations* and *detail-in-context visualizations*. In the following we will use the term *focus+context visualizations*, which is wide enough to encompass all the properties we will be discussing.

2 Related Work

Although the origin of focus+context visualization can be traced back to non-interactive distortion-based techniques for visualization of map data [14], the first computer-based interactive method was introduced with the *FISHEYE View* [8], more known as the *Generalized Fisheye View* [9]. This original fisheye notion was in fact a general interaction framework for information filtering according to the user's current point of interest in the material, rather than a specific visualization technique, and was shown to be applicable to various types of data, notably structured programs and tree structures. (Some confusion has been the result of several other techniques using the term "fisheye", and currently fisheye visualization is often more closely associated with distortion-based techniques that give the graphical impression of the fisheye-lens of a camera.) In connection with the Generalized Fisheye View, important concepts such as the *Degree of Interest* (DoI) function and the *Level of Detail* (LoD) were introduced.

Another early interactive example of focus+context visualization was the *Bi-Focal Display* [29], where a graphical focus+context display was applied to a calendar display, introducing distortion in the horizontal dimension. A somewhat similar technique, the *Perspective Wall* [20], used a 3D perspective to achieve the same effect. The *Document Lens* [24] developed the concept further by combining a perspective view with a magnifying-glass effect to give combined detail and overview presentation of a document. Other techniques that use various forms of distortion to display two-dimensional images or maps include the *Graphical Fisheye View* [25] and *Rubbersheet View* [26], and forays have been made into extending such techniques to three dimensions [7]. *Flip zooming* [11] was developed to visualize sequentially ordered material, and it has been used for visualizing documents [12] and hierarchically ordered image collections [13]. Techniques developed specifically for visualizing graphs and hierarchies include *Hyperbolic Trees* [18], the *Continuous Zoom* [2], and *Cone Trees* [23].

Among papers seeking to classify or formalize focus+context techniques, [19] is probably the most widely cited. It gives an overview of the various techniques and provides a unifying theory in the form of a rubber-sheet analogy. [10] introduced *Space-scale diagrams* as a framework for analysis of multi-scale (or zooming) interfaces, and showed that such diagrams could also be used for describing focus+context techniques. So-called *Non-Linear Magnification Fields* [16] have been introduced as an abstract representation of distortion-based magnification techniques, and these have since been more generally applied to the problem of detail-in-context visualization [17]. [28] introduced several dimensions of transformation, *X, Y, Z,* and *W,* where the W-transformation corresponded directly to the Generalized Fisheye View.

3 The Focus+Context Visualization Process

3.1 Levels of Representation

When describing information visualization, it is often sufficient to describe the underlying data, how the data is represented and what manipulation or interaction this representation will allow [6, 30]. Manipulation can be either manipulating the data itself, or, if the visualization is interactive, manipulating the way in which the data is presented. Focus+context visualizations can also be described in this way. However, we argue that it is useful to describe a focus+context visualization as a *second-level visualization,* i.e. a visualization of a visualization.

To clarify this, consider the rubbersheet metaphor as described in [19]. Here, a focus+context visualization is compared to a sheet of rubber that has an image of some sort printed on it, e.g. a map or document. The rubbersheet is tied up in a rigid frame, representing the fixed size of the screen. Magnification of a certain area can then be achieved by stretching part of the sheet, and due the limited space available within the frame, other areas will shrink correspondingly. According to our distinction, we would say that manipulating a second-level visualization corresponds to manipulating the rubbersheet itself. Manipulating the first-level visualization, however, would correspond to some manipulation of what information is actually printed on the sheet.

This distinction is important, since in many cases it might be interesting to be able to perform manipulations at *both* levels of visualization. Separating the levels in this way will make the different types of interactivity clearer, and will also make it easier to account for how we can combine different focus+context visualizations with different types of information visualization techniques. In the following, some examples will be given to illustrate this.

3.2 Example 1: Structured high-level computer program

Here, the data consists of a sequence of code that represents a computer program. One way to visualize and interact with a program would be to show it as a succession of lines, indented according to their place in the program structure, in which the user can scroll up and down. The program might also be represented as uniformly sized pages of text, which the user can switch between (this would reflect the way the program would look when printed on a laser printer and might be useful when making changes according to comments written on a print-out). We might also isolate the various components of the program, such as functions and data structures, and show these as nodes in a hierarchically ordered tree; this would represent the inherent hierarchical structure of the program.

On any of these visual representations, we can then apply a focus+context visualization technique. In the case of lines of indented text we might choose to use the Generalized Fisheye View [9]. If we have text separated into uniformly-sized pages, we might use the Document Lens [24] or the Zoom Browser [12]. If we choose to have the program represented as a set of hierarchically ordered objects and functions, we might want to use the Hyperbolic Tree Browser [18] or Cone Trees [23].

Considering the interaction that might be possible in the system, users should of course be able to manipulate the data itself by making changes in the code; these changes will directly affect the data, and will be reflected in the first-level visualization as changes in the text, indentation, hierarchical structure, etc. But users can also manipulate the focus+context visualization by means of changing the focus, increasing or decreasing the degree of magnification, etc. These changes are occurring in the second-level visualization, and will not change the actual data, only the way it is shown to the user.

3.3 Example 2: Geographical elevation data

When creating a geographical model of a certain area, the data can be described as a number of data triplets, with the two first values representing coordinates in the plane, and the third component representing the altitude. A common way to represent this type of data is to create a graphical map in two dimensions, where gray-scales or colors indicate the altitude. In some cases, however, it might be useful to use a table of the underlying numerical values, perhaps for working with the data in a spreadsheet application. Alternatively, we might create a fully 3-dimensional representation of the data, which could be rotated and viewed from different angles.

A 2-dimensional map is the most common representation used for this kind of data in focus+context visualization, as it is suited to for many distortion-based techniques, such as the Rubbersheet View [26] and the Graphical Fisheye View [25]. A very different, but still valid, type of focus+context view can be given of the tabular data with a technique such as the Table Lens [22]. In the case of a fully three-dimensional representation there may be a natural focus+context effect in the use of perspective: the parts that are close to the point of view will be more into focus than parts further away. However, for a more generalized focus+context view of 3-dimensional data, methods such as those presented in [7] might be used.

Considering the interactivity, if the map data is only to be viewed as-is, users might only interact with the information at the focus+context, i.e. second, level of visualization, by changing the focus and magnification, etc. However, if the user is going to change the data in some way, say do some manual corrections to the survey values, this interaction will take place at the first level, and be directly reflected in the table, map or other underlying visual presentation.

4 A Formal Description

We will now describe the focus+context visualization process in a more formal manner.

4.1 Visualizations

Any information visualization starts with a set of data, i.e. the information to visualize. A visual representation of this data set – or some set of data derived or constructed from this set – can be constructed based on the values or inherent structures of this data. Let us define this information visualization as:

IV ([D], V, I)

Here, **IV** is some form of information visualization in which **[D]** is the set of underlying data, **V** is how the data is presented visually, and **I** the interactivity or manipulation possible in the information visualization.

We must here distinguish between two different ways of manipulating **IV**. If **I** affects **[D]**, we can use **IV** according to **I** to manipulate the underlying data set **[D]**. This would for instance correspond to making changes to the data in a spreadsheet or a word processor. A different mode of manipulation is when **V** is affected by **I**, i.e. when a user can manipulate **IV** in order to change the way **[D]** is presented. An example of this is the case with visual information searching through dynamic queries [27], where the user can customize the visualization to show certain aspects of the data, without making any changes to the underlying data set.

4.2 Second-level Visualizations

If we instead of using **[D]** in the formula above insert some information visualization **IV**, or rather, a structure of visualizations, **[IV]**, we will have a second-level visualization, **IV’**:

IV’ ([IV], V’, I’)

Here **IV’** is the new second-level visualization, **[IV]** is the underlying set of information visualizations, **V’** is the second-level visual component, and **I’** is the interaction or manipulation possible in this visualization. This formula will now enable us to import any information visualization set **[IV]**, with its constraints **V** and **I** for how the structure can be visualized and changed, and apply any suitable new visualization and interaction method to this representation. Of course, in the same way as certain representations are only suited to certain types of data, **[IV]** may have to meet some constraints in order to fit into a certain second-level visualization **IV’**.

4.3 Focus+Context Visualization

We will now describe focus+context visualization as an instance of a second-level visualization **IV’**. It will take any set of information visualizations **[IV]** as its input, given that **[IV]** is compatible with the focus+context visualization technique in question. We apply a visual presentation component **V’** and some interaction **I’** that reflects the focus+context method chosen. As we incorporate some underlying information visualization **[IV]** rather than some data set **[D]**, we can focus on the aspects of **V** and **I** that are unique to focus+context techniques.

Interaction. The most notable aspect of interaction in focus+context visualization is the ability to select a focus and have the presentation changed accordingly. A convention introduced in [8] is to call the point (or rather, object) in focus **‘.’** (dot). Now, we can ask how other objects in the underlying visualization **[IV]** are related to **‘.’**: given a **‘.’** $\in$ **[IV],** how important is another object **x** $\in$ **[IV]**? According to the same convention, this can be termed the *Degree if Interest*, **DoI.** In order to answer this, we have to describe the relation between **‘.’** and **x,** or rather, the “distance” between **‘.’** and **x**. The distance will depend on how closely the two objects are related to each other, but also of the individual properties of **x**. In [8] the function *Level of Detail* was used to establish a measure of this distance. The level of detail of an object **x** reflect where in a hierarchical structure it belongs; objects belonging to higher levels (i.e. more abstract) are said to have a lower level of detail, and hence they are more important when providing a general context. Let us use:

W (. , x)

Where **W** is the weighted distance between **'.'** and **x**, or in other words the importance of **x** given **'.'** (where **'.'** and **x** ∈ **[IV]**).

However, there are other ways of controlling how closely related two objects are as well. We might for instance let the user link objects to each other, ensuring that whenever one of them is in focus, the other one will be brought forward as well. We might also allow for other ways of weighting the objects besides using their position in a hierarchy, making it possible for individual objects to have an independent "importance factor" associated with them. Furthermore, we might want to use a tool similar to the focal length on a camera, controlling how big the difference between the focus and context should be. At one extreme the use of such a tool would imply that nothing but **'.'** is seen, and at the other that there is no difference between **'.'** and the rest, i.e. a maximal and a minimal difference between **'.'** and the rest of **[IV].**

Visualization. Given that we know which object is in focus, and how important the other objects in **[IV]** are in relation to it, we can create a visual presentation. As the available resources are limited, some constraints have to be met. This makes it useful to introduce a threshold function, **T**. **T** depends on the size of the screen, **s,** its resolution, **r,** and the computational resources, **c,** available (at least if real-time interactivity should be possible). Hence we have:

T (s, r, c)

The threshold function **T** gives a value of how close an object will have to be to **'.'** in order to be visualized. In order to determine whether a certain object **x** should be visualized or not, the weighted distance **W (. , x)** is compared with **T**:

W (. , x) > T

However, in some focus+context techniques objects are never excluded, meaning that **T** is not used to determine whether **x** should be visualized or not (or, alternatively, that **W (. , x) > T** for every **'.'** and **x** ∈ **[IV]**).

W (. , x) can also be used in order to determine which, if any, transformations of **x**'s underlying visual presentation **IV** (which is presented according to **V** in the underlying representation) should be made, e.g. distortion or scaling. For example, **x** can be given an amount of space on the screen proportional to its distance to focus as defined by **W(. , x)** in which case **V'** can be a simple scaling of the image produced by **V**. **W** can also be used to determine where to display **x** in relation to **'.'**, representing **W** with actual distance between objects on the screen.

Besides functions depending on **'.'** and **W(. , x)**, transformations of the underlying representations and rules for screen layout can also be applied. For instance, structural aspects of **[IV]** can be used to determine where on the screen a certain object should be placed. If the objects in **[IV]** are ordered sequentially, say, as the pages in a book, we might want them to be ordered in the same way on the screen, whereas if **[IV]** is presented hierarchically, we would want the focus+context presentation to reflect this accordingly.

5 Applying the Framework

Having defined the formal framework, we can now use it to describe some of the examples presented earlier.

Considering the first example, the structured computer program, we have one set of data that is the code being edited, which we can term **[C]**. We can then choose to have some interactive representations of it: a line-based representation, or one based on discrete uniformly-sized pages of text, or one based on a hierarchically ordered set of components. Let us call them $\mathbf{CV_L}$ (line-based code visualization), $\mathbf{CV_P}$ (page-based), and $\mathbf{CV_H}$ (hierarchical), respectively. Examining the components **I** and **V** of each representation, we see that the visual component **V** in the first case is a long sequence of lines of code, in the second it is a number of sequentially ordered pages of equal size, and in the third **V** is a number of differently sized chunks of code each representing a logical unit of some sort, presented in a tree structure. Similarly, in the first case **I** allows us to move up and down in the sequence of lines; in the second, it will allow us to switch back and forth between discrete pages of code; and in the third, it allows us to navigate the hierarchical structure of the program. If we term these components $\mathbf{V_L}$ (line), $\mathbf{V_P}$ (page) and $\mathbf{V_H}$ (hierarchy), and $\mathbf{I_L}$, $\mathbf{I_P}$, and $\mathbf{I_H}$, respectively, we have the following formulas:

$$\mathbf{CV_L = IV\ ([C], V_L, I_L)} \quad \textbf{(line-based visualization)}$$

$$\mathbf{CV_P = IV\ ([C], V_P, I_P)}\quad \textbf{(page-based)}$$

$$\mathbf{CV_H = IV\ ([C], V_H, I_H)}\ \textbf{(hierarchical)}$$

We can now insert these representations into a focus+context visualization. Common for all of these will be that the **I** component will allow the user to move the focal point, **‘.’**, in some way. In the Generalized Fisheye View, this will be through focusing on a single line; in the Document Lens and The Zoom Browser we can focus on a single page; and the in the Hyperbolic Tree and Cone Tree, we can move a certain point in the hierarchy into focus. These interactions, which we can term $\mathbf{I_L}$**’** (line-based interaction), $\mathbf{I_P}$**’** (page-based) $\mathbf{I_H}$**’** (hierarchical), respectively, correspond directly to the interactive components of the first-level representation.

The visual component **V’** in the various cases has these properties: In the Generalized Fisheye View, only certain lines of code will be shown according to their degree-of-interest, with most detail being shown nearest to the focus; this we will term $\mathbf{V_L{'}_{DoI}}$ (line-based degree-of-interest view). In the Document Lens the pages surrounding the focus will be distorted according to the combined perspective and optical metaphor used, but will keep their relative position. This we can call $\mathbf{V_P{'}_F}$ (page-based focus+context view with fixed position). With the Zoom Browser, all surrounding pages will be shrunk to the same size, and re-arranged sequentially according to the browser’s left-to-right, top-to-bottom convention; this we call $\mathbf{V_P{'}_S}$ (page-based view with sequential position). Finally, in the Hyperbolic Tree Browser and Cone Trees, the act of focusing on one component will affect how the other components are shown according to their place in the hierarchy, so that components farther away in the hierarchy will be less visible, with close objects more visible. This we will call $\mathbf{V_H{'}_H}$ (hierarchical view based on hyperbolic geometry) and $\mathbf{V_H{'}_{3D}}$ (hierarchical view based on 3D-perspective), respectively.

We can now describe any of the focus+context applications in this example in a formal way. For instance, the Generalized Fisheye view (let us call it **GF)** becomes:

$$\mathbf{GF = IV'\ ([CV_L], V_L{'}_{DoI}, I_L{'})}$$

In the same way, the Hyperbolic Tree (**HT)** used on our hierarchically ordered program becomes:

$$\mathbf{HT = IV'\ ([CV_H], V_{H'H}, I_{H}')}$$

Using Cone Trees (**CT)** on the hierarchical ordering gives us a similar formula:

$$\mathbf{CT = IV'\ ([CV_H], V_{H'3D}, I_{H}')}$$

The other focus+context examples can be constructed according to the same principles.

We can also do some novel combinations. Say that we want to apply the Hyperbolic Tree view to a set of uniformly-sized sequential pages. Since the only structure we have access to is the discrete pages in sequential order, $\mathbf{I_P}$, we will have to base the interaction on this, but the visualization can still be done using hyperbolic geometry. Let us call this new Hyperbolic Tree variant $\mathbf{HT_P}$:

$$\mathbf{HT_P = IV'\ ([CV_P], V_{H'H}, I_{P}')}$$

Since the visualization is designed to reflect a hierarchical structure, $\mathbf{HT_P}$ might not be of much practical use, but the important point is that such novel applications can be constructed in this framework.

Similarly, returning to the map example, we may term the underlying geographical data **[G]**. If we choose to represent it as a static 2-dimensional map, **M,** we may have a visual component $\mathbf{V_{M2D}}$ (2-dimensional map) but no interaction component (resulting in **I** being empty). We can then apply, say, a Rubbersheet View to this map, with the visual component being that of rubbersheet deformation, $\mathbf{V_R}$, and the interactive component being that of rubbersheet interaction, $\mathbf{I_R}$. The Rubbersheet View (**RV**) visualization of a static map would then be:

$$\mathbf{RV = ([M], V_R, I_R)}$$

Where $\mathbf{M = ([G], V_M, I)}$, and **I** is empty. However, we might want to have an interactive rather than a static map as first-level representation of **[G]**. For instance, if we want to have a zoomable map, being able to zoom in on certain parts for further visualization in the Rubbersheet view, we may have $\mathbf{M_Z = ([G], V_M, I_Z)}$, if $\mathbf{I_Z}$ is the zooming interaction and $\mathbf{M_Z}$ is the resulting zooming representation of the map. This can then be inserted in the Rubbersheet view, resulting in a new variant:

$$\mathbf{RV_Z = ([M_Z], V_R, I_R)}$$

An interesting scenario would be to add some more complex interaction to the first-level representation, say a set of dynamic query sliders [24] to facilitate advanced visual data retrieval. We would then insert the interaction $\mathbf{I_{DQ}}$ for the dynamic query searching, getting the resulting dynamic query-based map visualization $\mathbf{M_{DQ}}$. By applying a Rubbersheet view we would then get a focus+context application which included dynamic query searching of the map data:

$$\mathbf{RV_{DQ} = ([M_{DQ}], V_R, I_D)}$$

This might in fact be quite a useful application, since it will combine an advanced visual query method with the detail and overview supported by the Rubbersheet. Thus, the formal system has been shown to handle both existing focus+context applications, and novel combinations of first- and second-level visualizations.

6 A Software Package Supporting the Model

As we have seen, it is possible to generate different focus+context visualizations given the same underlying representation, or to apply the same focus+context visualization to a number of different representations, by varying the parameters described in the theoretical framework. This property of the formal description makes it suitable for implementation as a general software platform. We have constructed such a software package, to support the creation of focus+context visualizations of information visualizations consisting of sequentially ordered discrete visual objects. The reason for this choice of underlying visualization is that the package grew out of our work with flip zooming [11, 12], which was developed specifically for this type of visualizations. However, the implementation of a general software package has allowed us to implement some quite novel variations of the original flip zoom concepts.

6.1 A Discrete Focus + Context Software Package

The package was constructed using the Java Abstract Window Toolkit [1]. It is based on two types of Java classes: *f+c (focus+context) components* and *f+c containers,* corresponding to **IV** and **IV'** respectively. An f+c component is based on a standard Java window component, with the added functionality needed to interface with a focus+context visualization. In terms of the formal description presented above, components must provide ways to facilitate event handling related to the interaction **I'** given by a higher-level visualization **IV'**. The **V** and **I** portions of the components provide the painting of the component on the screen, and the handling of input from keyboard and mouse, for instance to facilitate manipulation of the underlying data set **[D]**.

The f+c components are stored within f+c containers, in the same way as **[IV]** is used in **IV'**. An f+c container is a Java subclass of the f+c component class, meaning that it inherits the properties of the component and must facilitate the same functionality. An advantage of this is that it is possible to insert an f+c container into another f+c container, making higher-order visualizations possible. Further functionality is needed in order to support the focus+context visualization; most notably, the containers interaction portion **I'** has to allow for sequential transversal and the random access of focus objects.

The visualization **V'** consists of two parts: The *f+c layout manager* and the *f+c visualizer.* The layout manager, which handles how the components are placed on the screen area, can be implemented according to a number of different strategies, giving rise to a number of different presentation styles. It determines the size and position of the components and provides methods for how to change the layout when setting, changing and losing focus, or when objects in **[IV]** are inserted or removed during execution. The actual drawing of the components is done by the f+c visualizer, which has access to the different visualization functions **V** in the underlying visualizations in **[IV]**.

6.2 Examples of different implementations

We have used the software framework to implement a number of sample applications. In the following, we will briefly describe some of these,

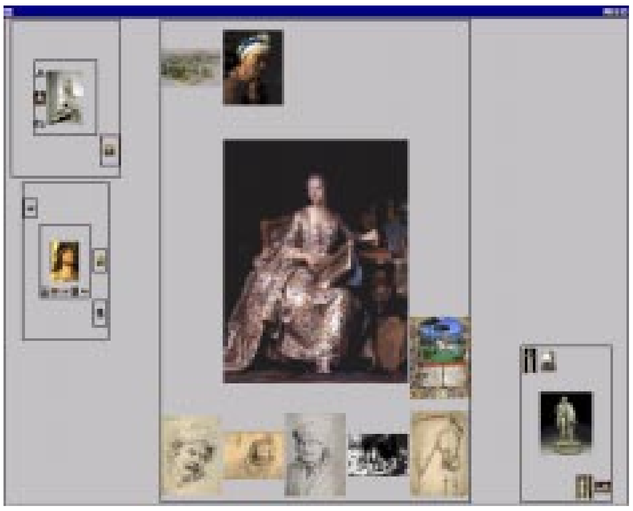


Fig. 1. The Hierarchical Image Browser.

focusing on how **IV** and **IV’** are related to each other. (More details on the applications can be found in the references.)

The Hierarchical Image Browser. The *Hierarchical Image Browser* [13] was designed to explore the possibilities of using hierarchies to present large image sets in a structured way (see **Figure 1**). The hierarchies might for instance reflect the way art is exhibited in a museum, i.e. being placed in different rooms, sections and floors according to the types of paintings. The images in the set **[IV]** were ordered into containers **IV’** according to their placement in the hierarchy. Further, these containers were ordered in higher level containers **IV’’**, **IV’’’,** etc., according to the hierarchical structure. This application shows how the general software framework allowed us to insert focus+context visualizations into higher-level focus+context visualizations, thus reflecting the general nature of the theoretical framework.

The Digital Variants Browser. Developed as an aid to literature researchers, the *Digital Variants* application [4] presented several versions of one text to facilitate comparative studies (see **Figure 2**). The application accommodated a number of document variants **IV,** each of

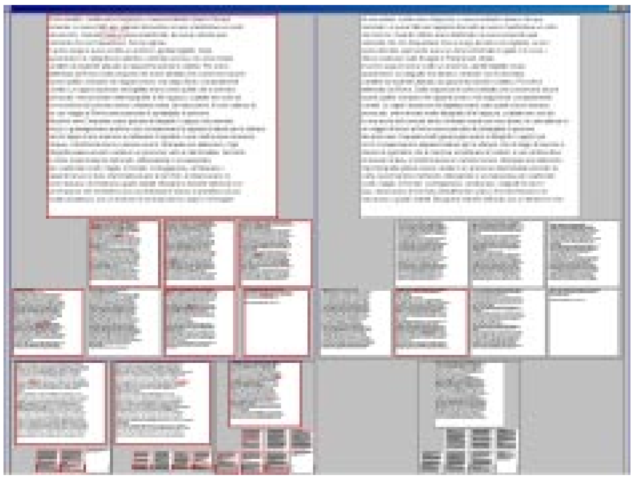


Fig. 2. The Digital Variants Browser; a total of six documents are shown, two are in focus.

which was presented in a focus+context display **IV'**. This set of focus+context visualizations **[IV']** was then visualized in a third-level focus+context visualization **IV"** of slightly different sort, namely one that allowed for two simultaneous foci, facilitating the comparison of two texts. This application shows how we could use the software framework to create second- and third-level focus+context visualizations with slightly different interactive and visual properties.

The WEST Browser. The WEST browser, a *WE*b browser for *S*mall *T*erminals [5], was developed for use on small mobile devices, such as Personal Digital Assistants (see **Figure 3**). Due to the limitations in display area (160 x 160 pixels) and computational power, both the space factor **s** and the computational factor **c**, put constraints on the visualization. To solve these problems, webpages were pre-processed in a number of steps to create a suitable structure **[IV]**. First, a web page was stripped of banners and divided into a number of small chunks, *cards*, each which would fit into the allowed screen space. The cards were then ordered in a hierarchical structure with no more than seven children to any node. All images in the original web page were scaled to the appropriate size and saved in the representation **[IV]**. Further, each of the cards was analyzed in order to find links and keywords. These were

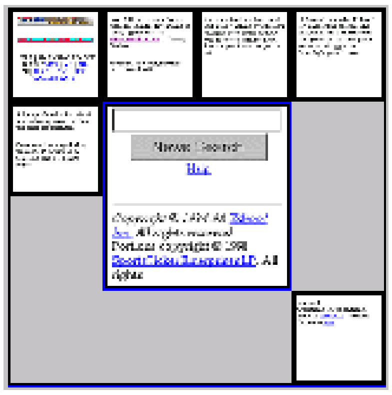


Fig. 3. The WEST Browser allows for several different views of the same web-page source.

used as complementary structures of the webpage in **[IV]**. Thus, the preprocessing delivered three sets of **[IV]**: one based on the graphical look of the cards, one based on the extracted keywords and one based on the links.

The interface **I'** of the WEST browser facilitated navigation between the different levels of cards representing one webpage, but also the traditional functionality **I** associated with a web browser, such as the ability to follow links and use a history list. The user could also switch between three views: normal webpage, keyword view and link view, thus visualizing different components of **[IV]** in the same higher-order visualization **IV'**. This application shows how the framework allowed us to construct a complex interactive visualization of several different underlying visualizations.

7 Discussion and Future Work

In this paper, we first presented arguments for separating focus+context visualizations into first- and second-level visualizations, supported by some intuitive examples. We then presented a formal framework for describing properties of such aggregated visualizations and the relations between them. This enabled us to describe our initial examples in a formal way, thus validating the formal framework. We showed that the

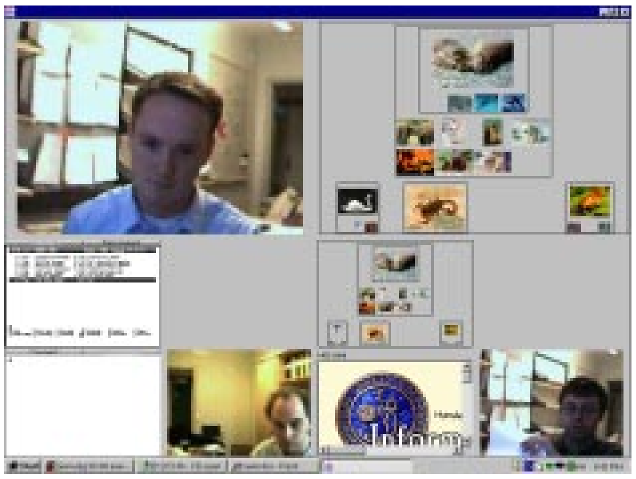


Fig. 4. The Focus + Context Desktop, incorporating several different applications.

framework allowed us to construct some novel combinations of first- and second-level information visualizations. We also described some work with a general software package based on the formal framework, including example applications that uses hierarchies of focus+context visualizations and multiple underlying visualizations.

We can now see that according to our formal description, any **IV** that fulfils the constraints posed by **IV'** can be incorporated into **[IV]**. This means that we can incorporate any information visualization **IV** into any higher-level visualization **IV'**. This opens a lot of interesting possibilities: there is for instance nothing to stop us from applying several focus+context visualizations **IV, IV', IV'',** etc. to each other. As we saw with the hierarchical image browser and the Digital Variants browser, this can in fact be a very useful technique for combining different types of views or building a hierarchical visualization**.**

In the software package, we also have the possibility of using different types of applications within a f+c container as long as they fulfil the specified criteria for being a f+c component. One example of such an application is the *Focus+Context Desktop* (see **Figure 4**), which incorporates any application displayed in a Java window, including web browsers, web-cameras, file directory browsers and telnet clients, into a common workspace based on focus+context visualization (similar systems include [3, 15]). Future work should include evaluating such systems, as well as further experiments with nested focus+context visualizations, and applications that have heterogeneous types of underlying visualizations

The framework given in this article is not limited to focus+context visualizations, and it should be possible to use it to describe and construct many other types of interesting higher-level visualizations. Similarly, it should be possible to construct a software framework that supports other types of visualizations apart from focus+context techniques. (As we have seen, the Java language is quite suitable for the construction of such software.) However, we need to better understand the properties of the visualization components, (**V, V',** etc.) and the interaction components (**I, I',** etc.). In particular, if we could isolate the necessary properties required for a certain higher-level visual component **V'** and interactive component **I'** to be compatible with the lower-order **V** and **I**, we will be able to state more clearly whether a certain combination of visualizations is likely to be practically useful or not. For instance, in the example section, we gave only an intuitive motivation for why Hyperbolic Trees might not be well suited to visualizing sequential data; if such relations could be expressed more formally, the usefulness of the framework should be increased quite significantly.

If extended in such a way, the framework might allow us to better explore the properties of novel visualizations even before they are implemented. It might provide answers to questions such as: What focus+context visualizations are best suited to a specific underlying visualization? How can different visualizations be combined in a focus+context visualization? How does the interactivity of a underlying visualization affect a focus+context visualization and vice versa? Our hope is that by making the distinction between different levels of visualization explicit, and by introducing a formal system that supports this notion, new possibilities within the design space of both focus+context techniques and information visualization in general will become available.

8 Acknowledgements

The work presented here was part of the project *Effective Display Strategies for Small Screens,* funded by SITI, the Swedish Institute of Information Technology. We are grateful to our project partners at Ericsson Radio Systems, Telia Research and the Swedish Institute of Computing Science, and to our colleagues at the Viktoria Institute.

9 References

1. Arnold, K. and Gosling, J. *The Java™ Programming Language,* Second Edition, Addison-Wesley, 1998.
2. Bartram, L., Ho, A., Dill, J., Henigman, F. The Continuous Zoom: A Constrained Fisheye Technique for Viewing and Navigating Large Information Spaces. *Proceedings of ACM UIST '95,* pp. 207-215, ACM Press, 1995.
3. Bederson, B., Hollan, J. Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. *Proceedings of ACM UIST '94,* ACM Press, 1994.
4. Björk, S. and Holmquist, L.E. The Digital Variants Browser: An explorative tool for literature studies. *Proceedings of Computers, Literature and Philology,* Edinburgh, UK, 1998. (To appear)
5. Björk, S. and Redström, J. An Alternative to Scrollbars on Small Screens. *Extended Abstracts of CHI '99,* ACM Press, 1999.
6. Card, S.K., Mackinlay, J.D and Shneiderman, B. Information Visualization. In Card, S.K., Mackinlay, J.D and Shneiderman, B. (eds.) *Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers, San Francisco, California, pp. 1-34, 1999.
7. Carpendale, M.S.T., Coperthwaite, D.J. and Fracchia, F.D. Extending Distortion Viewing from 2D to 3D. *IEEE Computer Graphics and Applications,* July August, 1997.
8. Furnas, G.W. The FISHEYE View: A New Look at Structured Files. Bellcore Technical Report, 1981. Reprinted in: Card, S.K., Mackinlay, J.D and Shneiderman, B. *Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers, Inc., San Francisco, California, 1999.
9. Furnas, G.W. Generalized Fisheye Views, *Proceedings of CHI '86,* pp. 16-23, ACM Press, 1986.
10. Furnas, G.W and Bederson, B.B. Space-Scale Diagrams: Understanding Multiscale Interfaces. *Proceedings of CHI '95,* ACM Press, 1995.
11. Holmquist, L.E. Focus+Context Visualization with Flip Zooming and the Zoom Browser. *Extended Abstracts of CHI '97,* ACM Press, 1997.
12. Holmquist, L.E. The Zoom Browser: Showing Simultaneous Detail and Overview in Large Documents. *Human IT,* vol. 2 no. 3, ITH, Borås, Sweden, 1998.
13. Holmquist, L.E. and Björk, S. A Hierarchical Focus + Context Method for Image Browsing. *SIGGRAPH '98 Sketches and Applications,* ACM Press, 1998.

14. Kadmon, N., and Shlomi, E. A polyfocal projection for statistical surfaces. *Cartograph,* J. 15, 1, 36-40, 1978.
15. Kandogan, E., Shneiderman, B. Elastic Windows: Evaluation of Multi-Window Operations. *Proceedings of CHI '97,* pp. 250-257, ACM Press, 1997.
16. Keahey, T. and Robertson, E.L. Non-Linear Magnification Fields. *Proceedings of IEEE Visualization '97, Information Visualization Symposium,* IEEE Press, 1997.
17. Keahey, T. The Generalized Detail-In-Context Problem. *Proceedings of IEEE Visualization '98, Information Visualization Symposium,* IEEE Press, 1998.
18. Lamping, J., Rao, R., Pirolli, P. A focus+context technique based on hyperbolic geometry for viewing large hierarchies. *Proceedings of CHI '95,* ACM Press, 1995.
19. Leung, Y.K, Apperley, M.D. A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Transactions on Computer-Human Interaction,* vol. 1 no 2, pp. 126-160, 1994.
20. Mackinlay, J. D., Robertson, G. G., Card, S. K. The Perspective Wall: Detail and Context Smoothly Integrated. *Proceedings of CHI '91,* pp. 173-179, ACM Press, 1991.
21. Plaisant, C., Milash, B., Rose, A. Widoff, S., and Shneiderman, B. Lifelines: Visualizing Personal Histories. *Proceedings of CHI '96,* ACM Press, 1996.
22. Rao, R. and Card, S.K. The Table Lens: Merging Graphical and Symbolic Representations in an Interactive Focus+Context Visualization for Tabular Information. *Proceedings of CHI '94,* ACM Press, 1994.
23. Robertson, G.G., Mackinlay, J.D. and Card, S.K. Cone Trees: Animated 3D Visualizations of Hierarchical Information. *Proceedings of CHI '91,* ACM Press, 1991.
24. Robertson, G.G., Mackinlay, J.D. The Document Lens. *Proceedings of UIST '93,* pp. 101-108, ACM Press, 1993.
25. Sarkar, M. and Brown, M.H. Graphical Fisheye Views. *Communications of the ACM,* vol. 37 no. 12, pp. 73-84, 1994.
26. Sarkar M., Snibbe, S.S., Tversky, O.J. and Reiss, S.P. Stretching the Rubber Sheet: A Metaphor for Viewing Large Layouts on Small Screens. *Proceedings of ACM UIST '93,* pp. 81-91, ACM Press, 1993.
27. Shneiderman, B. Dynamic Queries for Visual Information Seeking. *IEEE Software,* 11(6), 70-77, 1994.
28. Spence, R. A. taxonomy of graphical presentation. *INTERACT '93 and CHI '93 conference companion,* pp. 113-114, ACM Press, 1993.

29. Spence, R., Apperley, M. Data base navigation: an office environment for the professional. *Behavior and Information Technology,* vol. 1 no. 1, pp. 43-54, 1982.
30. Tweedie, L. Characterizing Interactive Externalizations. *Proceedings of CHI '97,* ACM Press, 1997.

WEST: A Web Browser for Small Terminals

Staffan Björk, Lars Erik Holmquist and Johan Redström, *PLAY*

Ivan Bretan, *Telia Mobile*

Rolf Danielsson, *Telia Research*

Jussi Karlgren and Kristofer Franzén,
Swedish Institute for Computer Science

Abstract. We describe **WEST,** a **WE**b browser for **S**mall **T**erminals, that aims to solve some of the problems associated with accessing web pages on hand-held devices. Through a novel combination of text reduction and focus+context visualization, users can access web pages from a very limited display environment, since the system will provide an overview of the contents of a web page even when it is too large to be displayed in its entirety. To make maximum use of the limited resources available on a typical hand-held terminal, much of the most demanding work is done by a proxy server, allowing the terminal to concentrate on the task of providing responsive user interaction. The system makes use of some interaction concepts reminiscent of those defined in the Wireless Application Protocol (WAP), making it possible to utilize the techniques described here for WAP-compliant devices and services that may become available in the near future.

Keywords. Hand-held devices, web browser, proxy systems, focus+context visualization, text reduction, flip zooming, WAP (wireless application protocol)

1 Introduction

The World Wide Web (WWW) currently consists of about half a billion pages, offering users a vast range of informational resources. However, these pages are almost exclusively designed for use with desktop computers, i.e. computers with large high resolution screens, powerful processors, and an abundance of primary and secondary storage.

Parallel to exponential growth of the web during the last few years, digital mobile telephony has evolved to become a basic commodity in the United States and many parts of Europe. Particularly in the Nordic countries, penetration can be as high as 60% (Finland). The world's largest manufacturers of mobile phones predict that there will be 1 billion mobile telephones in use in five years time. Currently, mobile communication is still mostly synonymous with voice telephony, but this is almost certain to change pending new mobile data communication technologies being deployed, increasing data speeds and improving usability. In particular, this development should be viewed in the context of network technologies such as GPRS (General Packet Radio Service), allowing for data speeds in the range of 115 kbps and service technologies such as WAP (Wireless Application Protocol) which sets an industry standard for web-like, interactive applications for use with mobile telephones.

The work presented here focuses on this encounter between the WWW and mobile telephony, and more specifically on the need to provide gateways between mobile technologies and existing web resources. Although mobile terminals require specially designed formats for optimal usability due to the constraints of the user environment, it is not likely that all information available on the web will be translated into these format in advance. Thus, there is need for some kind of automatic on-the-fly transformation of existing web content to mobile formats, in order not to shut mobile users out from the bulk of web resources.

In dealing with this issue, the crucial problem is not as much a lack of bandwidth (which the new network technologies are dealing with) or the conversion from one mark-up language to another, but rather developing techniques for the adaptation of information to the usability requirements of mobile terminals. Innovative use of techniques for information filtering and information visualization seem to be a fruitful approach in dealing with this problem, as such techniques deal with issues that are part of the problem of providing information on small mobile devices.

New ways are needed to present web resources and to navigate among and within web pages, which is the target domain of the work described here. The constraints on information presentation posed by small terminals made it necessary to combine several different strategies in order to achieve a sufficiently compact presentation. In different fields of research, several techniques for creating compact representations have been developed. In WEST, techniques from computational

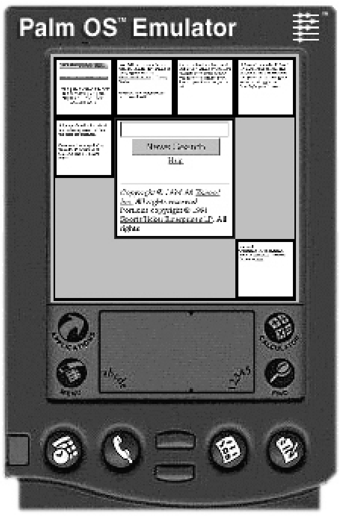


Fig. 1. The WEST browser on a simulated Palm OS™ display.

linguistics and information visualization were combined. The original web-pages were compressed both in terms of their linguistic content by means of text reduction, and in terms of their visual presentation, and were then presented to the user by means of focus+context visualization.

The rest of the paper is organized as follows: First we give a brief overview of the WEST system and its components. We then give a background in related work, required for implementation of the system. A detailed example, where we see how a user may interact with the system follows. We then describe each of the components of the system in further detail, and give an account of an early user test of the system. Finally, conclusions and future work are discussed.

2 The WEST Browser

WEST (**WE**b browser for **S**mall **T**erminals) is a web browser specifically designed for use on hand-held devices with limited resources (see **Figure 1**). Although most of the actual implementation was done in

Java running on a standard PC, in order to achieve a realistic simulation of the conditions of mobile computing we based the design of WEST on the capabilities and limitations of popular PDAs, such as the 3Com Palm™ line of hand-held computers. Such a device would typically have a small touch-sensitive black-and-white or grey-scale screen with a resolution of about 160x160 pixels, a memory of about 0.5-4 MB, a processor running at about 10-20 MHz, and no provision for traditional keyboard input. These capabilities have proven to be quite adequate for the tasks which such devices are currently required to perform, but are of course far below the specifications of any current desktop computer. The challenge was to work within these limitations but still provide the user with a workable browsing experience, and in the process attempt to overcome the navigation problems that would typically occur on a small terminal.

Our solution consisted of two parts:

- *A proxy server* (running on the user's ISP server), that would take a standard HTML page and transform it in real-time into a format suitable for browsing on small screens
- *A client application* (running on the hand-held terminal), that would allow the user to view and interact with the web pages as provided by the proxy server

The reason for letting the bulk of the processing of the HTML pages be done on the proxy server rather than by the terminal was to relieve the comparatively under-equipped terminal of resource-intensive tasks, thus allowing it to concentrate its resources towards providing responsive user interaction. Furthermore, by stripping away unwanted information on the server rather than on the client, a saving on bandwidth might also be made.

The proxy processing was comprised of several stages:

- *A chunking stage,* where an HTML page was divided into a number of smaller pages, or cards, which were then collected into groupings, or decks
- *A text reduction stage,* where a set of keywords summarizing each card were extracted from the text
- A *link extraction stage,* where all the hyper-links on each card were extracted

The resulting cards, with supporting keywords and links, were then passed to the client. The client application would then provide the following display modes:

- *Thumbnail view:* Here, a focus+context visualization comprising miniature views of the cards (or top-most card of each deck) was provided
- *Keyword view:* Here, rather than presenting thumbnails, the keywords extracted from each card were presented
- *Link view:* Similar to the keyword view, but rather than displaying keywords, this view showed the links available on each card

(A *pure text view,* showing only the text with no images or formatting, was not included in this prototype but could be useful in some situations and might be added later.)

Each view allowed the user to zoom in completely on a card, providing a fully readable view of the content. The user interacted with the views using the *flip zooming* focus+context visualization technique [16], through which the system provided an overview of the material with simultaneous access to the individual cards.

3 Related Work

In designing the system used in the WEST prototype, previous work from several different research areas were applied: Proxy systems to provide intermediate formats of the web pages; text reduction algorithms to find suitable keywords in the pages; and information visualization techniques to display information on the limited screen space available. Some properties of WAP, the Wireless Application Protocol, were also considered.

3.1 Wireless Application Protocol (WAP)

WAP [32] is a de facto standard for providing Internet-based content and services to wireless devices such as cellular telephones, and requires resources to be coded in a dedicated mark-up language called WML (Wireless Mark-up Language) adapted to the limitations of such devices. Although the WEST architecture is not specifically designed to work in conjunction with WAP, there is potential for interesting synergies when it comes to user interaction.

Firstly, the concept of deck (approximately corresponding to a page in WWW, i.e. a single resource transfer) and card (sub-unit of deck, i.e. a single display object) in WAP lends itself very well to visualization using flip zooming. The overview mode captures the collection of cards, i.e., the deck, whereas the zoomed view corresponds to viewing an individual card. Because of this nice correspondence, we have adopted the WAP terminology of cards and decks in the WEST system, although WAP protocols are not currently used in WEST.

Secondly, given a PDA type of device with WAP capabilities, a WEST browser could readily be converted into a WAP browser, i.e. processing WML instead of HTML. However, when it comes to mobile phones with small text-only displays, this is of course a completely different issue. In this setting, the simplest way of using a WEST browser would be to navigate in zoomed-only mode (i.e. without the context overview). A more advanced solution would involve creating overviews through keyword or link-extraction.

3.2 Proxy Systems

The notion of using a proxy server to mediate between the Internet and thin clients is well established [9, 23]. A proxy of this kind can have many functions: coding and conversion of protocols; filtering, compression and conversion of information, etc. The WEST proxy could be tailored to include protocol functions, but the work presented here focuses on the information handling aspects of a web proxy for mobile devices. By removing unwanted or unnecessary information, and by compressing and restructuring (chunking) the information, it can be made to better suit the usability demands of mobile terminals. When it comes to information compression, we can distinguish between *lossy* and *non-lossy* compression. This distinction is normally applied to images [9, 23], but in the context of WEST we will be dealing with lossy text reduction in the shape of text summarization techniques.

WEST has in common with the *Top Gun Wingman* browser for the 3Com Palm™ PDA [10] the basic principle of hiding complexity (such as HTML parsing and analysis) in the proxy server to off-load the hand-held device. The idea of saving screen real estate by using text compression has been put to use in another proxy-based system known as the *Digestor* [2]. Proxy systems can also be used as support systems in “surgical” extraction of information from WWW and other sources, providing semi-automatic conversion of such pre-determined content into format suitable for thin clients, such as WML or Web Clippings in the Palm VII™ PDA. *Panama* from Oracle [26] is an example of such a system, which converts HTML and other formats into XML, from which selected WML fragments can be generated by means of stylesheets.

It should be noted that the work presented here does not take a stand as to whether pre-authored content (e.g., a WML source) or automatically converted and filtered content (e.g., HTML–>WML or HTML to simplified HTML) is the strategically correct way to produce services for the user of wireless handheld devices. We content ourselves with the observation that there will be a demand from mobile users for accessing arbitrary web-based resources, particularly in the initial deployment period where dedicated mobile services, WAP-based or not, will emerge slowly. Initially, the range of services and content for mobile use will be limited, since information providers will be reluctant to invest in parallel coding of content. Gradually, this will change (particularly if systems such as Panama are used, which allow for re-use of existing web resources), but there will always arise situations where users want to access material not pre-adapted to dedicated mobile formats. The WEST approach tries to address the needs of such users.

3.3 Text Reduction

For the keyword view, a text had to be summarized into a few words. We call this technique *text reduction,* to distinguish it from traditional text summarization. The major challenge for traditional text summarization techniques is two-fold: understanding which regions of a text bear the most pertinent information, and cobbling those bits of information together into a coherent summary. In the case of small screens, the space requirements are more demanding, which actually makes the task somewhat easier. Coherence will not be an issue, since the aim will be to extract a small number of information-bearing terms from the text, making the task closer to the field of index term selection than that of text summarization.

Index terms are typically selected based on term frequency as originally proposed by Luhn [24], selecting suitably frequent terms to represent a document. However, the most frequent words in a text are usually form words, which bear little or no topical information (“is”, “and” and the like). These words must be filtered out either through the application of a judiciously composed stop list or through the application of estimates of term specificity [30]. These are terms that occur in all documents in a document base and have no indexing power; terms that occur in few documents are more useful to that end. Typically, the two measures are combined, in a standard “tf.idf” formula (e.g. [29]). This was the basis for the keyword extraction algorithm used in our application.

3.4 Viewing Web Pages on Small Screens

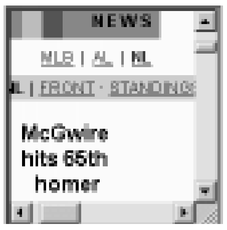
Although personal digital assistants and other hand-held devices have been available for a number of years, the problems associated with user interface design for small terminals have only recently started to attract attention from the human-computer interaction research community [18, 25]. While many general principles for human-computer interaction also apply to small terminals, they can not always be taken for granted, and to simply transfer interaction components from desktop computers will often lead to unexpected problems [15].

Earlier research in information visualization techniques have focused mainly on maximizing the use of screen space on ordinary computer screens. A number of *focus+context techniques* have been developed to give users access to simultaneous overview and detail. General focus+context visualizations techniques such as the *Generalized Fisheye View* [11] or techniques developed for text documents, such as *SeeSoft* [7] or the *Document Lens* [27], might be adapted to the WWW. General zooming or multi-scale interaction techniques which have also been used for visualizing web pages include *PAD++* [1], *Cone Trees* [28], *Hyperbolic Trees* [22] and *Elastic Windows* [19], and techniques developed specifically with the web in mind include the *WebBook* and the *Web Forager* [6], *Zippers* [5] and *CZ Web* [8].

Although most of the techniques above have been developed for use on traditionally-sized screens, some of them might feasibly be adapted for use on small screens. However, many of these techniques have advanced requirements in the form of computational resources for performing smooth graphical transformation and providing responsive interaction, and while they may often have proved useful on desktop machines, hand-held devices such as those on which the WEST system are intended to be used, are currently for the most part not capable of any advanced visual calculations. The focus+context technique *flip zooming* that was used in this project was also originally developed for ordinary screens, but because it is not very resource-intensive it has proven possible to transform it to smaller devices. For ordinary screens, it has previously been used for visualizing web sites [14], and has been generalized to handle hierarchical material such as hierarchically ordered image collections [17]. As part of the WEST project, we have evaluated flip zooming as an alternative to scroll bars on small screens [3].

4 Interaction in WEST

To give a better idea of how the WEST browser works, we will now give a detailed account for how a user may interact with the system. This will take the form of a complete interaction scenario, with an illustration for each screen the user will see.



The example page viewed in a traditional browser on a 160x160 pixel display

As our example, we have used a page reporting baseball news at the Yahoo Sports site. The page was comprised mostly of text – 319 words, or about 1500 characters. There were 15 links to other pages, plus a banner advertisement and a search function. As the figure above will attest, viewing this page on a traditional browser on a 160x160 pixel screen presents serious problems. Only a very small part of the page would then be available at any time, giving almost no clues to the size or context of the material.

4.1 Flip Zooming in WEST

The interaction in WEST is based on *flip zooming*, a tile-based focus+context visualization technique. Flip zooming allows users to navigate a data set consisting of sequentially ordered discrete objects, e.g. images or pages of text. One object is in focus, the other objects provide the context. Users can move the focus backwards or forwards in the data set, or select any visible object as focus object by pointing at it. Users can also zoom in further on an object, allowing it to occupy the whole screen. Objects are ordered in a left-to-right, top-to-bottom fashion, so that any object that is after another object in sequence will be placed to the right and/or below the preceding object.

Earlier user studies of flip zooming applications [4] have indicated that users may become confused if thumbnails and focus objects are allowed to change their positions on the screen, or if the display is too packed with information. For this reason, we limited the maximum number of objects on the display at any one time to seven, which allowed us to keep the focus object at the center of the display with sufficient room to display the context objects at a reasonable size. It may seem that in some cases we are not using the available screen estate to the maximum, but this is a conscious trade-off to provide a clearer and more easily-understood display.

In WEST, some objects on the display are in fact representations of several objects, since they represent the top-most card of a deck. In this case, when zooming in on such a card, a user would be presented with a view of all the cards in the deck, which could then be navigated as usual. In this way, the user is in fact navigating a hierarchy comprised of decks and cards. In the example this hierarchy is only one deck deep, but there is no reason why it could not be more complex.

4.2 Interaction Example

We will now follow a user interacting with the sample page using the WEST browser. The user wants primarily to read about her favorite Chicago Cubs player, Sammy Sosa, and possibly chat about his exploits with other supporters.[1]

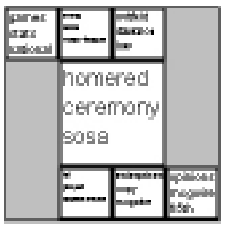
1. Thumbnail view, whole page, with first card in focus

1. Initially in WEST, the viewer is presented with the *thumbnail view,* which gives an overview of the whole web page in flip zoom format. Here, each card is presented as a thumbnail image, not large enough to be readable, but still giving a sense of the overall nature of the page – e.g. image-heavy, text-heavy, many or few links, etc. The first card or deck is in focus, with the others presented as context. (Unfortunately, there is currently no clear visual indication of if a thumbnail represents a single card or a deck, something which might be addressed in future versions of the browser.)

2. Keyword view, whole page, with first card in focus

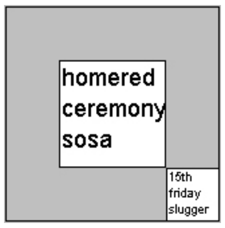
2. The user now chooses to switch to the *keyword view,* to see if she can locate some information about Sammy Sosa.

1. The authors know very little about the game of baseball, and apologize in advance for any errors in this account that may be spotted by more knowledgeable readers!



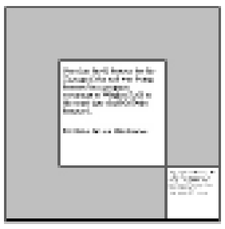
3. Keyword view, whole page, fourth card in focus

3. The keywords on the fourth card in the sequence indicate that Sosa is mentioned. The user focuses on that card. This can be done either by explicitly pointing at the card, or by moving the focus sequentially until the desired card is reached. (Since what here looks like a single card may in fact be the top card in a deck, user will actually often be navigating among decks in this manner.)



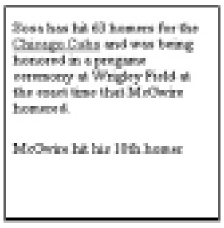
4. Keyword view, a deck open, first card in focus

4. The card in question is in fact a deck, consisting of two sub-cards in total. By zooming in on the visible card, the first card in the deck, the deck is opened and displayed. The keywords indicate that some kind of ceremony has taken place, involving Sosa and home-runs.



5. Thumbnail view, a deck open, first card in focus

5. The user now switches back to a thumbnail view of the deck, showing the original HTML formatting of the cards.

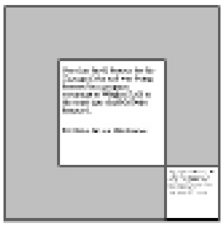


Sosa has hit 63 homers for the Chicago Cubs and was being honored in a pregame ceremony at Wrigley Field at the exact time that McGwire homered.

McGwire hit his 10th homer

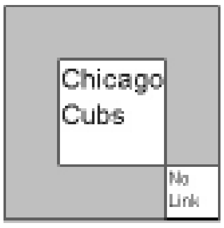
6. Thumbnail view, zoomed in completely on a card

6. The user zooms in completely on the first card in the deck and reads the text on the card. It is indeed interesting news about Sammy Sosa. Staying in this view, the user can now advance to the next or previous card in the deck (e.g. by pressing a specified button on the PDA or tapping on a portion of the card with the pen), to read the full story. (If the card on view happens to be the last in a deck, when advancing, the first card in the following deck will be shown.)



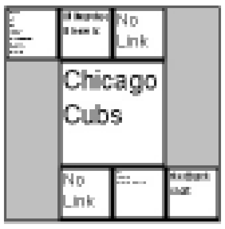
7. Thumbnail view, a deck open, first card in focus

7. The user now wants to chat with other supporters about this development. She zooms out again, returning to the overview of the deck.



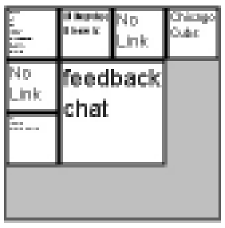
8. Link view, a deck open, first card in focus

8. The user now switches to the *link view*, since she is looking for a link to the chat page.



9. Link view, whole page, with fourth card in focus

9. Not finding the link she is looking for in this deck, she zooms out to reveal link view for the whole page.



10. Link view, whole page, with seventh (last) card in focus

10. She sees a link to the chat room on the very last card in the page, and focuses on that card. By clicking on the link while the page is in focus she will be transported to the chat-room page, meaning that the current web page/deck will be removed from the screen and the chat-room page/deck will be displayed.

5 Description of the Components

The components of the WEST system were designed as a number of modules that could be individually improved and expanded as the system was developed. In the following, we will describe each of these pieces separately.

5.1 Pre-processing, Including Card Chunking

Proxy servers for real-time pre-processing of web information to be accessed using a mobile terminal is a proven technique used for instance in current web services for palm-sized PDAs. In WEST, we made use of a proxy server to:

- Filter and reduce the contents of web pages in order to adapt them to the capabilities of the mobile browser (this would mean among other things to get rid of JavaScript, image maps, frames etc.)
- Convert the reduced web page into n sub-pages (cards), each of which can be readily presented on a mobile-sized display (e.g. 160x160 pixels). Cards are inter-linked to form a deck by arranging them into a suitable reading-order
- Produce alternative renderings of these cards corresponding to different levels of detail. Typically a card can be displayed in its full size, in reduced size and minimized. These alternative renderings are not necessarily derived from graphical reduction – in WEST, one alternative when reducing card size is to use automatic text summarization

Key element such as headings, paragraphs, pictures, tables etc. provide hints on how the original page is structured. These hints are used in the “chunking” of the page into cards, i.e. determining break-points for card creation. The maximum allowed size of a card is of course a limiting factor, which sometimes means that the information contained within a card’s minimal natural page-chunk cannot be presented without some modifications (for instance image or font size adjustments), or by splitting up the information into two cards. (For more information on the chunking algorithm see the appendix.)

The cards produced by the proxy are arranged into several decks linked together in the original reading order. Because of the limitations of the display, each deck was limited to seven cards, the maximum that could comfortably be displayed using the flip zooming variant we had chosen. If a page consisted of more than 49 cards (seven decks with seven cards each) some of the decks would in turn have to contain sub-decks of cards, creating a deeper hierarchy.

5.2 Extraction of Keywords

To extract the keywords that were to represent each card, the method chosen had to be suitably general to handle any kind of material. Since there is no way in advance to tell what type of web page a user will be browsing, the system should be equally at home at whatever topic it was subjected to, including general news, sports, entertainment, and so on. It would be feasible to allow the creator of a page to specify which keywords are most relevant, but this would require that pages were specially constructed for the purpose of this system, and as mentioned, the intention was to give users access to all pages of the web without any prerequisites.

Our text reduction algorithm relies on the fact that typically several texts or text chunks will be compressed and displayed simultaneously. The text chunks are all short, approximately thirty words. The word tokens in the text chunks are tabulated for frequency, after the application of a stop list filter of form words. Each chunk is represented by a list of words sorted by frequency. These raw frequency counts are then modified by inverse document frequency [30] – each word will have its raw frequency count divided by a factor depending on the number of texts it has been found in. Thus, if a text has two words with equal frequency, where one occurs in three texts and the other only in the text at hand, the latter term will be weighted higher.

The text reduction procedure thus disfavors words that are evenly spread out over the chunks at hand, and aims at representing each chunk by as unique words as possible, in that given context of chunks. Words that occur disproportionately often in a given chunk, compared to other visible chunks, are favored above the more generally frequent words. But words with high frequencies that occur in many texts are not discarded. They are set aside and used to generate a header for the group of text chunks under consideration, and can be used for hierarchical reduction of the entire group, although this is not done in the current version of WEST. A group of chunks can be reduced to words such as “baseball”, “scores”, “season”; individual chunks can be more finely reduced to “sosa”, “homered”, etc. As mentioned previously, taking advantage of these headers could be particularly fruitful when using small text-only displays where the contextual overview does not fit.

As it stands today, the algorithm does not make use of morphological analysis, thesauri or lexical categories, all of which would increase the reliability of the results. Adding surface level linguistic processing is a modular issue and can be done without a system redesign: there are several efficient general-purpose linguistic analysis components suitable for this purpose.

5.3 Link Extraction

To facilitate a view of which links were available on a given web page a simple link extraction procedure was created. This went through each of the cards constructed in the chunking processing and created a similar deck structure, but where the content of each card would only be the hyper-links.

5.4 Web Page Rendering

For the graphical presentation of the different cards, each individual card had to be rendered as if it were a web page. However, we were unable to write a full-scale web rendering engine within the constraints of this project. Instead, we used the rendering engine provided by the HotJava Web Browser [31] to produce an image of each card as displayed on a screen of the required size (160x160 pixels). The same images where also graphically compressed to intermediate and thumbnail size. These pre-rendered images were then used by the system for the graphical presentation.

5.5 Presentation and Interaction

Based on the flip zooming technique, the WEST browser presents each web page as a number of discrete objects, representing individual cards or decks of cards. The user navigates between different objects by using directional buttons or by directly choosing the object to focus upon with a pen or other pointing device. For sequential reading of a whole page, a user would generally switch to a full-screen view and then advance through the cards by pressing a designated “forward” button.

Each view in WEST only presents one level of the hierarchical structure of decks and cards that represent the web page. To move between levels, the user zooms in on the object in focus (usually by clicking or tapping with the pen on it) and will thus go one level deeper into the structure. To go up one level, the user clicks or taps on the “white space” between the objects. This navigation might also be facilitated by the use of “up” and “down” buttons, for moving up and down in the hierarchy, analogous to zooming out and zooming in.

When the user goes down one level in the hierarchy, the focus object takes over the whole screen space to show its content. If the current focus represents a single card, this card will be allowed to fill the screen completely to facilitate reading. In the case of the focus representing a deck, however, a view of all the objects in the deck is presented.

The system provides three different modes in which the material can be viewed: thumbnail, summary and link view. When switching from one view mode to another, the position of the focus in the hierarchical structure is maintained, enabling the user to navigate in a suitable view mode to locate a card, and then change to another mode (typically the thumbnail view) to actually view the card. In the prototype, the user switched between the different views by accessing a pop-up menu.

6 User Experience

To gain some insight in how the WEST prototype performed with inexperienced users, we performed a qualitative evaluation in which the prototype was compared with the HotJava browser [31]. It is important to note that the test was in no way intended as a “fair” comparison between two browsers, since the HotJava browser was not developed with the intention of being used on very small screens. Rather, the intention was to gauge novice users initial reaction to the WEST browser, and the other browser was provided as a reference point only.

A test group consisting of ten subjects, all expert computer users but with no experience of browsing the web on a PDA, were set a number of tasks to perform both in the WEST system and the traditional browser. The tasks consisted of finding specific items in the material, and in some cases required returning to a part of a page which had previously been visited. The tests were performed on a traditional computer screen, but both browsers were given the same screen size as a typical PDA to operate in, i.e. a window of 160x160 pixels.

A questionnaire given to users after the test indicated that they thought that the prototype provided a better overview than the HotJava browser, ranking it on average 3.40 points higher in this respect (5.30 vs. 1.90, standard deviation being 0.68 and 0.99 respectively) on a scale of 1 to 7 with 7 being the best. It also showed that users thought searching was easier with WEST than with HotJava, ranking it on average

2.25 points higher on the same scale (5.55 vs. 3.30 with standard deviation 0.90 and 0.95 respectively). However, it was also noted that the flip zooming interaction technique took some time to get familiarized with, providing some initial difficulties.

Although we did not collect any quantitative measures during this preliminary experiment, the positive reactions of the users did provide us with an indication that the ideas behind the system should be worth pursuing further.

7 Future Work

At the moment, the system can be improved primarily in the following areas: improving the chunking of pages; improving the techniques used for text reduction; and improving the means of interaction with the system to make it useful in various realistic situations. We might also consider the division of tasks between the proxy and the client. At the moment, most of the work is performed on the proxy to off-load the client machine as much as possible. With faster hand-held machines, there is no reason to believe that not more or maybe most of the information processing such as keyword- and link extraction could take place on the client rather than on the server.

The chunking process still leaves much room for improvement, since often the provided cards are not of the optimum size for the available screen space. Improving the chunking is difficult, however, since there will have to be a balance between producing chunks that are logically coherent to the user, and chunks that are of maximum size. To achieve maximum chunk size it is sometimes necessary to break the pages at inconvenient places, even breaking text in mid-sentence, but this should be avoided for the sake of the user. A more thorough analysis both of page structure and user behavior will be needed to improve this process. Also, integrating the chunker more closely with the actual rendering of the HTML pages would make the judging of available space much easier.

The text reduction algorithm as it now stands is very simple. It is based on well established and understood techniques from text indexing, which guarantees a predictable, stable, and somewhat mediocre result. There are two well known bottlenecks in this type of information access techniques: 1) we have too little knowledge of texts as texts to be able to answer the question of what a certain text is about, and 2) we have too little knowledge of what the text will be used for and why the user wants it. The second problem is somewhat less pressing for this specific application: we know that the text needs to be compressed, and we know what the context is, namely what else is being displayed at the same time. This knowledge we already utilize to some extent, since we are able to generate a header for the texts in view at any given time. The first problem is harder. Knowledge of texts is limited if we view texts as simple bags of words. In future work, we plan to utilize stylistic information [20] to reduce different types of text differently: a legal text might be reduced to a paragraph header, while a long-winded error message might be reduced to a generic icon. We intend to experiment with using text structure to tailor the chunking algorithm so that it will feed homogenous bits of text to the reduction algorithm (e.g. [13]). We might use language technology such as surface syntactic analysis [21] and text extraction techniques [12] to extract topical terms and other topical items such as names, links, or dates from the text segments. We are currently running a pilot project for multi-document summarization, to be able to impose a middle level of analysis: the idea is to collapse several texts into one short summary, whereupon that summary in turn can be reduced.

Finally, it might be possible to improve the interaction with the WEST system in certain usage situations. Using a pen to interact with a hand-held device is sometimes undesirable, since it requires the user to hold the device with one hand (or place it on a flat surface) and use the pen with the other. Essentially, flip zooming only requires four navigational actions to navigate a hierarchical data set (move the focus back, move the focus forward, zoom in and zoom out), and while the WEST browser requires additional input for switching among the different views, it is in many cases possible to use perform the majority of the navigation using only four buttons without relying on a pen. This might allow users to navigate with more precision and efficiency in some situations, and ideally it might even be possible to construct the system so that all navigational buttons were accessible using just one hand, thus freeing up the other hand for other tasks. This would make the human-computer interaction far more flexible, as there might be many situations when having one hand free would be beneficial: while talking in a phone, taking notes, etc.

8 Conclusions

Truly mobile web access will evolve along several paths. One path is the development of the “stripped-down” web, reminiscent of browsing with text-only browsers such as Lynx. The other extreme will result from miniaturizing standard computers into hand-held devices capable of handling the same resources as stationary machines. These paths will of course cross, and we will see combinations of dedicated mobile resources and advanced hand-held computers. No matter what, the restrictions of mobile terminals will always hold with respect to the usage environment. We believe work like WEST is important because it focuses on ways to enable advanced interaction on small devices, ways that are largely independent of the capabilities of both the network and the terminal.

By constructing the WEST system, we have shown how material on the World Wide Web can be made available for mobile users and others who are restricted to accessing the web from small terminals. By placing the major work-load on the proxy server, and by providing a novel combination of visualization and text summarization, existing web pages can already be made much more suitable for such devices. In the future, with the continued acceptance of hand-held devices and high-speed wireless network, browsing the web from a PDA or a mobile phone will be a common occurrence. In these cases, systems such as WEST may aid in making this a much more pleasurable and productive experience.

9 Acknowledgments

Thanks to the other members of the WEST project, Roberto Busso and Peter Nilsson, and to the anonymous UIST reviewers whose valuable comments helped improve this paper. This work was part of the project *Effective Display Strategies for Small Screens* within the *Mobile Informatics* research program, sponsored by SITI, the Swedish Institute for Information Technology.

10 References

1. Bederson, B., Hollan, J., Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. In *Proc. UIST '94*, ACM Press, 1994.
2. Bickmore, T.W. and Schilit, B.N. Digestor: Device-Independent Access to the World Wide Web. In *Proc. Sixth International World Wide Web Conference,* pp. 655-663, 1997.
3. Björk, S. and Redström, J. An Alternative to Scrollbars on Small Screens. In *Extended Abstracts of CHI '99,* ACM Press, 1999.
4. Björk, S. and Holmquist, L.E. Formative Evaluation of a Focus + Context Visualization Technique. In *Proc. HCI '98* (poster presentation), The British HCI Society, 1998.
5. Brown. M.H., Weihl, W.E., Zippers: A Focus+Context Display of Web Pages, in *CD-Rom Proc. WebNet '96,* Association for Advancement of Computing in Education (AACE), 1996.
6. Card, S.K., Robertson, G.G. and York, W. The WebBook and the Web Forager: An Information Workspace for the World Wide Web. In *Proc. CHI '96,* pp. 111-117, ACM Press, 1996.
7. Eick, S.G., Steffen, J.L. and Sumner, E.E. SeeSoft - A Tool for Visualizing Line Oriented Statistics Software. *IEEE Transactions on Software Engineering,* 18(11), 1992.
8. Fisher, B., Agelidis, M., Dill, J., Tan, P., Collaud, G., Jones, C., CZWeb: Fish-eye Views for Visualizing the World Wide Web. In *Proc. HCI International '97,* pp. 719-722, Elsevier, Amsterdam, 1997.
9. Fox, A., Gribble, S.D., Chawathe, Y. and Brewer, E.A. Adapting to Network and Client Variation Using Active Proxies: Lessons and Perspectives. *IEEE Personal Communications* (invited submission), Sept. 1998.
10. Fox, A., Goldberg, I., Gribble, S.D., Lee, D.C., Polito, A. and Brewer, E.A. Experience With Top Gun Wingman: A Proxy-Based Graphical Web Browser for the USR PalmPilot. In *Proc. IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98),* Lake District, UK, 1998.
11. Furnas, G.W. Generalized Fisheye Views. In *Proc. CHI '86,* pp. 16-23, ACM Press, 1986.
12. Grishman, R. Information Extraction: Techniques and Challenges. *Materials for Information Extraction (International Summer School SCIE-97),* ed. Maria Teresa, Pazienza, Springer-Verlag, 1997
13. Hearst, M. and Plaunt, P. Subtopic Structuring for Full-length Document Access. In *Proc. ACM SIGIR '93,* ACM Press, 1993.

14. Holmquist, L.E. Focus+Context Visualization with Flip Zooming and the Zoom Browser. In *Extended Abstracts of CHI ’97,* ACM Press, 1997.

15. Holmquist, L.E. When Will Baby Faces Grow Up? In *Proc. HCI International ’99,* 1999.

16. Holmquist, L.E. and Ahlberg, C. Flip Zooming: A Practical Focus+Context Approach to Visualizing Large Information Sets. In *Proc. HCI International ’97,* pp. 763-766, Elsevier, Amsterdam, 1997.

17. Holmquist, L.E. and Björk, S. A Hierarchical Focus + Context Method for Image Browsing. In *SIGGRAPH ’98 Sketches and Applications,* ACM Press, 1998.

18. Johnson, C. (ed.). *Proc. First Workshop on Human Computer Interaction with Mobile Devices*. URL: http://www.dcs.gla.ac.uk/~johnson/papers/mobile/HCIMD1.html, 1998.

19. Kandogan, E., and Shneiderman, B. Elastic Windows: A Hierarchical Multi-Window World-Wide Web Browser. In *Proc. UIST ’97,* pp. 169-177, ACM Press, 1997.

20. Karlgren, J. and Cutting, D. Recognizing Text Genres with Simple Metrics Using Discriminant Analysis. In *Proc. COLING 94,* Kyoto, 1994. (In the Computation and Language E-Print Archive: cmp-lg/9410008).

21. Karlsson, F., Voutilainen, A., Heikkila, J. and Anttila A. (eds.) *Constraint Grammar,* Berlin: Mouton de Gruyter, 1995.

22. Lamping, J., Rao, R. and Pirolli, P. A Focus+Context Technique Based On Hyperbolic Geometry for Viewing Large Hierarchies. In *Proc. CHI ’95,* ACM Press, 1995.

23. Liljeberg, M., Helin, H., Kojo, M., and Raatikainen, K. MOWGLI WWW Software: Improved Usability of WWW in Mobile WAN Environments, in *Proc. IEEE Global Internet 1996 Conference,* London, England, November 20-21, 1996.

24. Luhn, H. P. A Statistical Approach to Mechanized Encoding and Searching of Literary Information. *IBM Journal of Research and Development*, 1 (4) 309-317, 1957. Reprinted in *Luhn, H.P.: Pioneer of Information Science, selected works.* Claire K. Schultz (ed.). New York: Sparta, 1968.

25. Marcus, A., Ferrante, J.V., Kinnunen, T., Kuutti, K. and Sparre, E. Baby Faces: User-Interface Design for Small Displays. In *CHI ’98 Summary,* pp. 96-97, ACM Press, 1998.

26. *Oracle Project Panama, Connecting Oceans of Information.* Oracle White Paper, March 1999. URL: http://www.oracle.com/mobile/panama/panamawp.htm

27. Robertson, G.G. and Mackinlay, J.D. The Document Lens. In *Proc. UIST ’93,* pp. 101-108, ACM Press, 1993.
28. Robertson, G.G., Mackinlay, J.D. and Card, S.K. Cone Trees: Animated 3D Visualization of Hierarchical Information. In *Proc. CHI ’91,* ACM Press, 1991.
29. Robertson, S.E. and Sparck Jones, K. *Simple, proven approaches to text-retrieval*. Technical report 356, Computer Laboratory, University of Cambridge, 1996.
30. Sparck Jones, K. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation,* 28:1, pp. 11-20, 1972.
31. Sun Microsystems. *HotJava HTML Component.* URL: http://java.sun.com/products/OV_hotjavaProduct.html
32. WAP Forum, *Wireless Application Environment Overview,* February 3, 1999. URL: http://www.wapforum.org

11 Appendix: The Page Chunker

To divide a page of HTML code into a number of pieces or *chunks*, each suitable for displaying as a single full-screen card on a small display, a page chunking program was developed. It was based on an existing HTML parser (or more accurately, an SGML parser with a description of HTML's elements) written in Java by Richard M. Tobin and available at the following address:

```
http://www.cogsci.ed.ac.uk/~richard/ftp-area/
html-parser
```

First, the page chunker establishes a number of constants, such as the size of a card (e.g. 160x160 pixels), the typical width of a character, the height of a line, maximum number of lines that can fit on a card, and so on. It then reads a piece of HTML code from an URL and performs a number of operations depending on the HTML elements encountered. Operations include:

- Setting flags for elements that can not be split and/or that are suitable as break-points (e.g. H1-H6, HR, A, IMG)
- Reducing the value for the total remaining space on the card (e.g. IMG)
- Adapting the width of the HTML element to the maximum available (e.g. PRE, HR, TABLE)
- Adapting the total size of the HTML element (width and height) to the maximum available (e.g. IMG, APPLET, OBJECT)

Additionally, some tags are replaced with tag combinations that will be handled in a more predictable way during page rendering; for instance, the paragraph tag <P> was replaced with <BR>&NBSP;&NBSP;&NBSP;

The chunker also makes sure that no tags are left "open" on a card, e.g. an opening <H1> with no corresponding closing </H1>. HTML elements are then added to the card until it is full, or as close to full as the algorithm can manage, at which time a new card is started.

After creating the cards, a number of decks are created. The current design of the flip zooming display in WEST limits the number of cards simultaneously visible at any time to 7. For simplicity's sake the deck creation algorithm simply tries to create a maximum of 7 decks with as equal a number of cards as possible. The resulting HTML files are saved in a file structure corresponding to the decks (i.e. one directory for every deck) which can then be read by the WEST browsing component.

Pseudocode for the page chunker is as follows:

```
chunkPage (parameter: URL for HTML page)
    parse HTML page
    save away header (<head> ... </head>)
    chunkBody; (divides page into cards)
    collect cards into decks and create corre-
     sponding file structure

chunkBody (parameter: HTML element)
    if not (tag=skiptag)
    then
     modify tag if needed, and add the (starting)
      tag (e.g. <IMG>), including attributes
      (e.g. an image) to body
    else
     if (tag=<p>) then reduce available space on
      this card with no. of characters on a line
    for (all sub-elements)
     if (element=text)
      then
       addText; (add this text to the new card)
      else (i.e. element=tag)
       addTag; (add this tag to the new card)
    if not (tag skipped) then add finishing tag
     (e.g. </IMG>)

addText
    (divide until it fits)
   while (number of characters added so far +
    length of new string >= maxlength)
    if (tag can not be split)
    then
     add string to current body
     add finishing tags to all open start tags
      and finish this body
     add corresponding start tags to new body
    else
     check where it is suitable to break (at end
      of paragraph/sentence etc.)
     add what we can fit in to the current body
     add finishing tags to all open start tags
      and finish this body
     add corresponding start tags to new body
   if (text left)
    then add remaining text to current body
```

addTag

```
  if (break condition) (i.e. is this a tag that
   can cause the creation of a new card?)
   then
   if (available space on current card is less
    than 10% of maximum size)
    then
     add finishing tags to all open starting
      tags and finish this body
     add corresponding start tags to new body
  chunkBody; (continue chunking body until done)
```

Supporting Group Collaboration with Inter-Personal Awareness Devices

Lars Erik Holmquist, Jennica Falk and Joakim Wigström

Abstract. An *Inter-Personal Awareness Device,* or *IPAD,* is a hand-held or wearable device designed to support awareness and collaboration between people who are in the physical vicinity of each other. An IPAD is designed to supply constant awareness information to users in any location without relying on an underlying infrastructure. We have constructed one such device, the *Hummingbird,* which gives members of a group continuous aural and visual indication when other group members are close. We have used the Hummingbirds in several different situations to explore how they affect group awareness. These experiences indicated that the Hummingbird increased awareness between group members, and that it could complement other forms of communication such as phone and e-mail. In particular, we found the Hummingbird to be useful when a group of people were in an unfamiliar location, for instance during a trip, where no other communication support was available. We argue that IPADs such as the Hummingbird may provide important functions in modern work situations.

Keywords. Handheld CSCW, awareness, mobile computing, wearable computing, ubiquitous computing, augmented reality

1 Introduction

The work situation in a modern office can be very different from the setting of only a few decades ago. Many workers are not obliged to come to work at specified hours and spend their whole time working at the same desk; instead, they are often much more flexible in working hours and location. Employees may keep variable hours and much time can be spent in meetings, visiting customers, performing field work, or working from home (telecommuting). Some may also have several workplaces that they move between, e.g. an office and a lab. At the same time, spontaneous meeting and group discussions continue to be an important factor in work, especially in creative professions. While much communication can now be carried out remotely, via telephone or e-mail, informal face-to-face communication is still very important [13].

A problem with initiating face-to-face communication is the matter of simply knowing where people are. Much time can be spent looking for people in order to initiate communication or call a group meeting. This also makes it important to fully utilize the situations when people really *are* present. Many desktop-based groupware systems support informal communication by conveying awareness of other people's activities, through constantly streaming video images or other means. An illustrative example is the *Portholes* application [6], where video-images of members of a working group were transmitted to the participants' desktop workstations at 5-minute intervals, thus providing a continuous awareness of the activities of others. The recent success of a commercial awareness-promoting product, *ICQ* (http://www.icq.com), is an indication of the desire of people to be aware of the (online) presence of colleagues and friends. ICQ users can be notified whenever certain other users are online, and the system, which also supports the sending of short messages, is said to have more than 20 million registered users.

However, solutions such as ICQ and Portholes assume that people spend most of their time at a desktop workstation. If people start moving around, or working from several places, perhaps using laptop computers or Personal Digital Assistants (PDAs), such workstation-based solutions will not work for initiating face-to-face communication or group meetings. Beepers and mobile phones are tools that work regardless of where the user is situated, but these do not provide continuous awareness information. Using a beeper or a mobile phone requires an explicit action by the user, making their effect quite different from a continuously running application such as Portholes, and furthermore they usually only support communication between two persons, not larger groups. A call via mobile phone or beeper can also be disruptive, since there is no way of knowing what activity the person at the other end is engaged in, and for this reason many people are wary of using them when not absolutely necessary.

Based on these observations, we argue that there is a need for an awareness solution that combines the advantages of desktop-based awareness applications - constant, non-disruptive awareness - with the freedom provided with mobile devices such as beepers and mobile phones. We propose that a new class of IT devices, *Inter-Personal Awareness Devices,* or *IPADs,* be introduced as a solution to the problems outlined above. In the following, we will define the IPAD concept and describe our first experimental implementation of such a device, the Hummingbird. We will then describe some experiences of using the Hummingbird prototypes. Finally, we will summarize the findings and outline some future work in this area.

2 Inter-personal Awareness Devices

An IPAD works as a *contact facilitator* rather than a *mediator.* This means that the IPAD is used to help initiate a contact, but not for sustaining the actual communication. IPADs extend the range of awareness provided by our ordinary senses, so that for instance a user may know that a colleague is nearby even though he or she is not close enough to be directly heard or seen. Many groupware applications already perform the same function, but as mentioned they are tied to the location of a user's desktop workstation rather than to the user herself.

From the name, an IPAD can be defined as having the following properties:

- *Inter:* It utilizes the relationship between itself and other IPADs (such as the distance), thus taking advantage of the user’s inherent mobility
- *Personal:* It is personal, i.e. an IPAD is identified with its user, and carried or worn at all times when in use
- *Awareness:* An IPAD is used to convey awareness of others. It is not used for mediation of the actual communication that may result from this awareness
- *Device*: An IPAD is a self-sufficient *device,* not reliant on any infrastructure except that provided by the presence of other IPADs.

An important basis for the IPAD concept comes from the observation that informal communication may occur whenever people are in the *same* place, but that it does not necessarily matter *which* place they happens to be in. For instance, noticing and talking to a co-worker at a café may be as important as meeting her at the office. Therefore, any solution constructed with the intention of promoting informal communication should, if possible, be usable independently of the physical location.

IPADs do not depend on any installed infrastructure and thus the communication between IPADs will be inherently bi-directional. For such a system to work effectively, every IPAD must send and receive the same type of information, and devices should be able to enter and leave the system as users come and go. Thus, there can not be any single device that has a crucial function, because if the users move out of range the system would stop working if it depended on any one of the components to function. It would of course be possible to construct a special IPAD that only sends, or only receives, certain information, but this would be closer to a traditional tracking or surveillance system and would not fit our current definition of IPADs.

Note that the IPAD definition is quite open-ended when it comes to functionality; for instance, it does not specify what *kind* of awareness should be conveyed. Typical examples of awareness to convey might be that of a person's level of activity, her mood, if she is available for contact or does not want to be disturbed, what her current task is, etc. Such information may be either set by the user or inferred automatically through some method. Also, the definition does not specify how the awareness information should be presented; it may for instance be through subtle audio, lights, tactile displays, etc.

Apart from the practical function of facilitating communication, the use of IPADs can be comforting. They can be used to convey the sense that a user is not alone or cut off from the group, even when the other members can not be directly heard or seen. This might for instance be when the user is situated alone in her room or in a crowded public place with many unfamiliar people around. Our usage experiences have shown that this is an important function of our current prototype.

3 Related Work

There is increasing evidence that important workplace collaboration takes place at many other places than at the users' desks [2]. Despite this, applications designed to promote informal communication by increasing awareness have so far been primarily tied to a desktop computer or some other stationary display, although badges that tell a centralized system about the user's location have been developed (e.g. *Active Badges*, below). In the desktop-based *Portholes* system video images of the members of a group were transmitted at five-minute intervals to increase awareness of the others' activities [6]. *@Work* provided a combination of video snapshots and other awareness information, some of which was made available via the web [10]. Systems with the purpose of supporting informal communication and awareness but which are not based on video communication include *Peepholes* [8] and commercial services such as *ICQ* (http://www.icq.com).

Several devices have been developed that use similar working principles to our IPAD prototype, the Hummingbird, but we are not aware of any that can perform the same function. The *Lovegety* [9] is a commercial "ice-breaking device", intended to match users of the opposite sex, currently mainly available in Japan. The devices come in two different varieties, "male" and "female", and when two devices of different kind are close (ca. 5 meters apart) they emit a piercing sound. Additionally, users can chose the preferred type of interaction, e.g. "talk" or "karaoke", and a visual signal indicates when there is a match. The Lovegetys have become a commercial success, but since they only work in "pairs" and support a very short range, they are not very suitable for use as inter-personal awareness devices. A similar system in the research community, *GroupWear* [3] matched users' interest (as defined by their answers to a questionnaire) and gave a visual indication of how well two users' profiles matched. *Meme Tags* [4], another application evolved from the GroupWear concept, was used to spread "memes" - short ideas in the form of text, input by the users – in a social setting. GroupWear and Meme Tags do not rely on any infrastructure, but since they communicate using infrared light, they require users to be in direct view of each other, thus not extending the physical range of awareness.

The *Active Badge* system [11] located users in a building, relying on an infrastructure of sensors. The badges themselves do not communicate directly with each other, making it impossible to use them outside of buildings which have the required infrastructure, thus losing much of the flexibility of the IPAD concept. Another system with computationally augmented badges, *SmartBadges* [1], has been proposed for use as a matching system similar to the GroupWear, with user profiles stored on servers. Although these badges are able to communicate via the Internet, a (nearly) ubiquitous infrastructure, wireless gateways to the network still need to be in place, making the system in practice only useful in areas where these gateways exist.

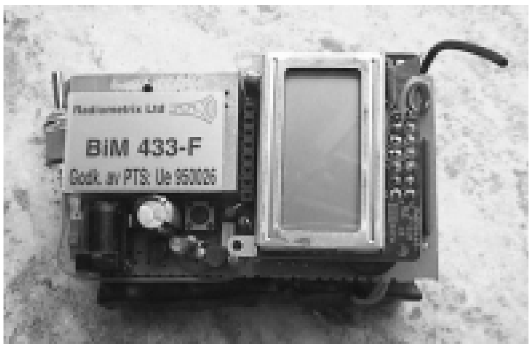


Fig. 1. The Hummingbird prototype

4 IPADs in Practice: The Hummingbird

To explore the concept of IPADs we have developed a prototype called the *Hummingbird.* The Hummingbird is an inter-personal awareness device that supports the *awareness of presence* between individuals in a group. It does this by providing users with an aural and visual indication of which other Hummingbird users are in the vicinity. The Hummingbird functions according to the following simple principles:

- A Hummingbird does not do anything on its own
- If two or more Hummingbirds belonging to users in the same group are close (currently less than roughly 100 meters apart) they will produce a sound – they “hum”
- In addition to the sound, a display supplies the identity of the other Hummingbird users in the vicinity (since there may be more than one user nearby at the same time)

In this way, the Hummingbirds can extend the awareness of presence between users even through physical obstacles like walls and closed doors. With Hummingbirds, it would for instance be possible for a group of users at the same workplace to be aware of each other’s coming and going, even if their individual workplaces are not located in such a way that they can always see or hear each other. This might be very useful in modern work situations, where people keep flexible hours and are not tied to a specific location, but where there can still be a need for both formal and informal meetings when the opportunity arises. Hummingbirds might also promote informal communication outside the workplace, since they do not rely on any infrastructure and can thus be used at all times.

After determining the desired functionality of the Hummingbird, we needed a way of releasing the concept through a prototype. Direct short-range radio communication between the devices seemed to be the best option. As a proof-of-concept, we built a first generation of prototypes. These were large, unwieldy circuit boards that did just one thing: when they got within a certain distance of each other, they produced a sound. These prototypes were too large to be comfortably carried but they did demonstrate the viability of the concept. The prototypes also gave us the opportunity to experiment with operating ranges, which would be very important for a successful realization of the Hummingbird concept. We soon found that standard radio components have such a high operating range that we had to artificially lower it, by measuring the strength of the signal and introducing a cut-off point. In this way, we could adjust the range of the prototype so that it was anything from a couple of meters to several hundred meters. After some experimentation, we settled on a range of about 100 meters suitable for triggering the Hummingbirds.

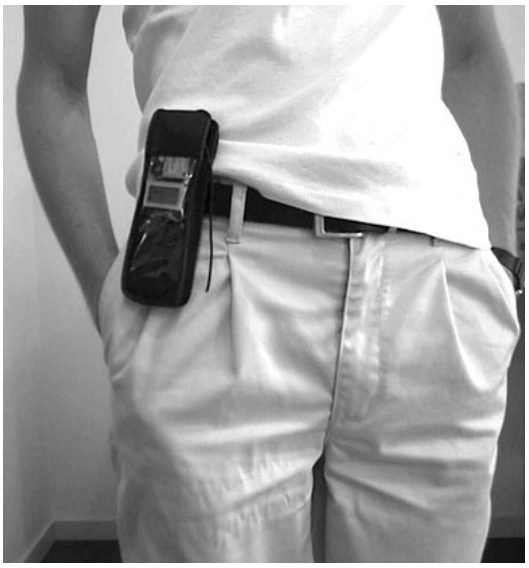


Fig. 2. The Hummingbird in carrying case.

We now had sufficient knowledge to build a second generation of prototypes, of which four were constructed for evaluations purposes. This device was now much smaller (see **Figure 1**); even with the addition of a power source it was no larger than a modern mobile phone. The prototype consisted of a circuit board with a micro-controller, a 2-by-8-character LCD screen, a miniature speaker, and a radio transceiver operating on the 433,92 MHz-band. It was powered by a set of rechargeable batteries, and would operate for about 10-15 hours on each charge. For convenience, we used a type of "holster" designed for mobile phones as carrying cases (**Figure 2**). Two switches were provided: one for turning off the sound without affecting other functionality; and one for turning the device on and off completely. Each Hummingbird was programmed to continuously transmit an identification code, while simultaneously listening for the codes of other Hummingbirds in the vicinity. It would have been possible to program the devices to transmit different codes to form several separate groups, but with only four prototypes we saw little reason to do it at this stage.

Since the size of the display forced us to keep the names of the devices short, we named them **a, b, c** and **d**. **Figure 3** shows a close-up of the Hummingbird display, in this case indicating that two other Hummingbird users can currently been found in the vicinity. (Some numerical information on signal strength and an arrow pointing to the letter of the latest detected device, that was made available for debugging purposes, can also be seen in the figure.) Users would be given a specific

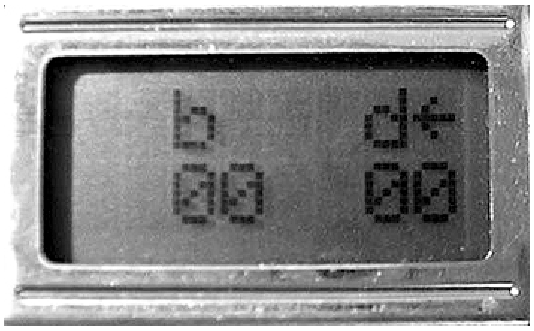


Fig. 3. The Hummingbird display. This device, **a**, has detected devices **b** and **d**. The arrow indicates that device **d** was the latest one to be detected; the numbers are an indication of signal strength, used for debugging purposes.

Hummingbird, so that when the devices were active, each user was associated with the name of the corresponding Hummingbird. We found that when properly briefed, users very quickly started to associate the letters with the person carrying the corresponding Hummingbird. The display proved especially useful when the surroundings were too noisy for the Hummingbird sound to be heard, or when users were in a situation where they had to turn off the sound so to not disturb other people.

5 Usage Experience

With four working prototypes, we were able to use the system in a number of different settings. Our aim was to attempt to incorporate the Hummingbirds into every-day situations, to find initial indications of how well they performed their intended purpose. It should be pointed out that the experiences described below were not intended as strict evaluations of the Hummingbirds. But although these results can at most be viewed as anecdotal evidence, they are examples of the system in real use, and we think they give some interesting indications of the effect IPADs may have on group communication.

In all situations described below, the user groups consisted of both people familiar with the Hummingbirds and novice subjects with no prior experience with the prototypes. We have grouped the experiences in two frameworks, which we call *familiar* and *unfamiliar* settings, respectively. We are aware that these definitions are not exhaustive and that there are many borderline cases, but they work reasonably well for the experiences that we will describe in the following.

Familiar setting. We define a familiar setting as an environment in which a person spends a significant amount of time, together with mainly the same group of people. In a familiar setting users will know most of the people around them, and keep in casual contact with them throughout the day. Typical examples are in the home – together with family and friends, and at an office or school – with colleagues or classmates.

Unfamiliar setting. We characterize an unfamiliar setting as a place or situation in which a person has rarely or never been. In an unfamiliar setting it is easy to get lost, physically or mentally, and for this reason it may sometimes require a great deal of effort to maintain contact with accompanying friends. Examples of this may include travelling abroad, visiting someone else's workplace, or at a large gathering of people such as a conference.

5.1 Familiar Setting

The office. In the building where the authors work, the offices are scattered across several floors. This makes it difficult to know when others are present, and people are often reluctant to make a trip through several floors or to the other end of a long corridor just to find out if someone is present. E-mailing or using the telephone can often help, but further complications are added by the fact that people may be at work but not in their rooms – they might be in meetings, tinkering in the lab, having coffee in the kitchen, etc. For these reasons, we thought that our own offices would provide an interesting opportunity for using the Hummingbirds in a familiar setting.

Four test participants carried Hummingbirds for a full working day, bringing them with them when they arrived at work in the morning and using them as they saw fit throughout the day. The only requirement was

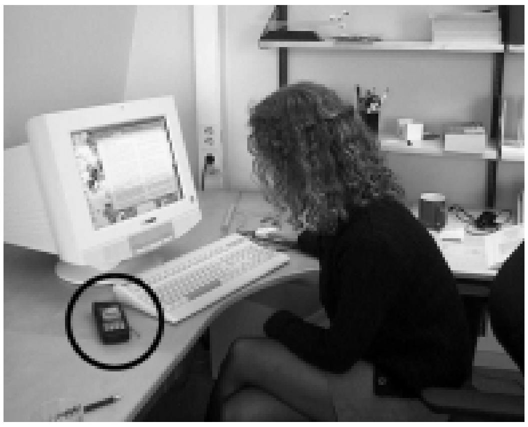


Fig. 4. In the office, the Hummingbird was mostly in the background of the user's attention.

to keep the Hummingbirds powered-on at all times. Despite working on the same project, the participants had offices located on three different floors in the building, a condition which we already knew decreased their awareness of each other. The strength of the Hummingbird signal was sufficient for users to know of the others' presence in the building, even when they were separated by several floors.

At the beginning of the day, users tended to actively monitor their Hummingbirds. There was a novelty value just in looking at the display to see who was present, and to take little walks around the house to see the effects. Soon, however, the first surge of interest waned and the Hummingbird would drift to the background of the user's attention. When in their offices working and concentrating on other things, the participants did not actively watch their Hummingbirds (see **Figure 4**). Only when there were distinct changes in activity would the Hummingbird be moved into the foreground of attention, for instance during hours of the day when people moved in and out of the building more frequently.

We found that in a familiar setting, the awareness information from the Hummingbirds was useful but not crucial. Users do not necessarily expect to meet the person they have established "Hummingbird contact" with. However, when they do need to find out if another user is present the Hummingbird allows them to instantly do so. Even if the other user is not in his or her office, just the ability to know if someone is in the building makes it easier to initiate contact, since there are only so many places in which to look. We did see some tendency for the four users to increase their informal interaction during the day, which we credit partly to the Hummingbird functionality and partly to the novelty of the experiment itself. Further long-term studies would be needed to confirm this effect.

From this study, we saw evidence that in a familiar setting Hummingbirds can work as a form of "calm technology" [11], which does not demand the user's undivided attention, but can be acted upon when needed. However, although we saw some effects on the interaction between users, these were nowhere near as strong as those found in the unfamiliar settings.

5.2 Unfamiliar Settings

We used the Hummingbirds in two settings that we classify as unfamiliar: At a large rock music festival, and at a major academic conference.

The rock festival. The annual rock festival in Roskilde, Denmark attracts around 80,000 visitors from all over the world. The event lasts for 4 days and offers a great variety of live music and other attractions. The festival is set in a very large and at times extremely crowded outdoor area, where it can be hard to maintain contact with companions. Here, we wanted to test the range and rate of contact of the Hummingbirds in a realistic outdoor environment and under realistic but semi-controlled conditions.

We considered it necessary to introduce some limitations in both time and space. The time limit was partly due to battery performance and the fact that once the batteries were discharged, we had no possibilities to re-charge them on-site. We also released we could not let users carry the Hummingbirds with them at all times, since the devices would probably break or disappear as soon as a user decided to watch a crowded performance or engage in some other typical festival activity. We limited the test to a pre-defined sub-section of the festival area, to increase the chances of establishing contact during the experiment. A roughly defined space, about 1,5 kilometres (one English mile) in diameter, was designated as our test area. The area was quite crowded in places and contained a variety of different attractions. The test was carried out during one afternoon, for a total time of four hours. Participants were told to wander around the test area and take notes of the time and place when a Hummingbird contact was established. Apart from this, the participants were free to use the information provided by the Hummingbirds in any way they wanted to.

After the test, some users said that the Hummingbird seemed to promote a feeling of “connection” that was quite unusual and not easy to explain or describe. The participants reported a clear sensation of connection with other users whenever their Hummingbirds established contact. When the Hummingbird contact was interrupted, and the other person disappeared out of range, the sensation of disconnection was just as evident. The participants often attempted to establish visual contact with other users that they knew were nearby, but soon found that a Hummingbird in itself does not always give enough information to locate a person. The users expressed occasional frustration when a person who was visible on the Hummingbird display could not easily be found. However, the Hummingbirds did give an indication whether it was worth looking for a person or not, which in most cases seemed to be enough, especially when contact was established in a fairly open space. Despite the occasional frustration, however, users were not certain that they wanted the Hummingbirds to give away any more information about distance and location, since they felt that this might be a breach in privacy.

Importantly, it was very evident that all of the participants enjoyed using the Hummingbirds! All participants talked positively about the experience, confirming that using the system has entertaining as well as practical benefits. Our conclusion from this experience was that Hummingbirds can work quite well in an outdoor setting where a group of people want to act independently, yet keep some contact with each other. We will avoid any attempts to explain the feeling of “connection” between users, and only note that it was quite evident that the Hummingbirds affected the communication in the group in a positive way. These findings were further substantiated by our final experience with the prototypes.

The conference. The annual ACM SIGGRAPH conference on computer graphics and interactive techniques is a combined academic conference and commercial exhibition, which attracts 30-50,000 visitors. In many ways it is surprisingly similar to a rock festival, since it too is a huge, sprawling environment with many things happening simultaneously. We brought three Hummingbirds to SIGGRAPH ’98 in Orlando, Florida (the fourth device being in a state of early retirement after the Roskilde experience). The test subjects were staying at three different hotels, but met regularly during the conference. They used the Hummingbirds extensively during the first few days of the conference, until the prototypes started to falter from the Florida heat and humidity.

As with the rock festival experience, the users found the Hummingbirds very fun to use, and started to rely on them to a surprising extent. When carrying a Hummingbird in the conference area, users would often check the display to see if someone else was close. When coming to a pre-determined meeting point, users would check their Hummingbirds to see if anyone else was there already. Although obviously a novelty even after several days of constant use, users soon allowed the Hummingbirds to complement other activities in a quite natural way.

The most interesting situation turned out to be when all three users attended a conference reception without having made any decision on which time to arrive. The first user who arrived at the reception could instantly determine that the others were not there yet, and rather than wasting time looking for his friends in the crowd, could concentrate on eating good food and chatting with other acquaintances. When the second user arrived, he immediately knew whom to look for, and found the first user almost immediately. When the third user arrived, she could see that both of the others were already there, and described the feeling as very comforting – “Oh good, I’m not alone!” – even though she had not yet seen any of her friends. We found that in this setting the Hummingbirds added noticeably to the enjoyment of the evening.

This test strengthened our impressions from the festival experience, in that it showed that Hummingbirds were both enjoyable and practical to use in an unfamiliar setting. Although the devices eventually broke down and we had to cut the experiment short, we felt that this experience indicated that long-term use of the Hummingbirds is viable. When the Hummingbirds started to malfunction and the experiment had to be abandoned, we found that the users genuinely missed the little “birds”!

5.3 Conclusions from the usage experiences

The most obvious conclusion from using the Hummingbirds was that although they seemed to have the potential to be useful in the office environment, users did not find them as immediately compelling to use in the familiar setting as in the unfamiliar settings. A reason for this could be that in the unfamiliar settings there was little other support for communication, even though such situations may be exactly those were users feel the need to communicate the most. With Hummingbirds, a comforting “link” to other people was created, which made the devices have a much higher short-term impact in this setting than when there was more communication support available and where the users felt more at home.

There was less initial enthusiasm for using the Hummingbirds in the office setting, but this test was very limited and we believe there is much potential for using IPADs in a variety of work situations. It is possible that if IPADs were to be used over a longer period of time, thus providing users with the opportunity to more fully rely on them in their day-to-day activities, they could prove a useful supplement to stationary awareness applications such as ICQ.

6 Future Work

We have already constructed a new generation of Hummingbirds, which is more robust and flexible. These are based around a pre-existing handheld computing device, the Nintendo GameBoy, which has been equipped with a radio transceiver and custom software. This construction, which is easier to produce and modify, will allow for some long-term and large-scale evaluations of the IPAD concept. We also intend to integrate the IPADs with traditional data networks, including the Internet, so that services such as ICQ may be complemented with IPAD functionality and vice-versa. Increased functionality will also allow for more advanced uses of the devices, including the specification of several different groups which can use the IPADs independently, allowing for much larger groups of users without conflict.

Formal evaluations will be needed to establish the potential usefulness of the Hummingbird and other IPADs, and to fully understand the effect that such devices can have on the communication in a group. These evaluations should include both close studies of groups of people

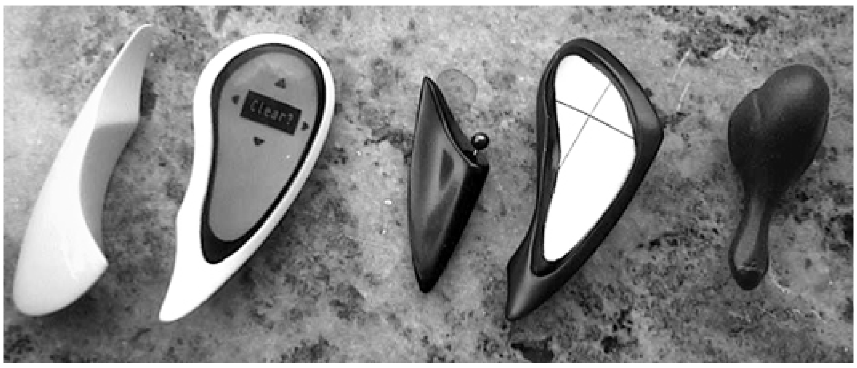


Fig. 5. Concepts for future IPADs (design by Jona J. Bjur)

using IPADs in unfamiliar settings, such as during travels, and long-term studies of the use of IPADs in familiar settings, such as an office or other workplace. Some kind of ethnographical method might be suitable for observing users in these settings, complemented with interviews to gauge their subjective reactions to the experience.

For a device that is to be carried and used at all times, the form factor is very important. We are currently working with an industrial designer and a jewelry designer to find new forms that would allow IPADs to be seamlessly integrated with personal clothing and accessories. **Figure 5** shows some proposed shapes. It is worth noting that these designs are noticeably smaller than any form of currently available mobile communication technology, something made possible by the fact that an IPAD should require very little user interaction apart from an on/off switch. In relation to the design aspect it might be interesting to work with other modalities than sound and vision, in particular haptics. Novel concepts for haptic communication such as *HandJive* [7] and *inTOUCH* [5] might provide inspiration for new forms of IPAD interfaces.

In a broader perspective, it will be important to examine how the use of IPADs might change the way we work and communicate. Will constant awareness information lead to "techno-stress", or will people learn to turn their IPADs off? If IPADs become popular, the development of their usage will most probably mirror that of the mobile phone, which is still in the stage of becoming naturally integrated in the daily life. A convergence of PDAs, mobile phones and IPADs may be a likely future for mobile work.

7 Conclusion

We think that it is important to open up our thinking about awareness in collaborative work, and move from traditional desktop-based applications to the mobile solutions provided by IPADs and other handheld CSCW devices. Since people will work in different places and the line between work and social life will probably blur even more in the future, the solutions that are provided should be flexible and not just tied to a specific workplace. We believe that the IPAD concept represents an avenue worth pursuing when continuous awareness of others is needed, both in traditional settings such as an office and in mobile settings.

IPADs also provide solutions to some of the privacy and security issues that arise from other similar solutions, such as Active Badge, since they are not reliant on any centralized information processing and there is no persistency of the location information they provide.

Our experience with the Hummingbird prototype shows that IPADs can be useful in a variety of situations. In our preliminary experiences, Hummingbirds were more appreciated when used in unfamiliar situations such as trips and conferences than in traditional office situations. This may be in line with the changing nature of work and the increased mobility that technology is making possible, and we think that it is quite natural that our experiences indicated that the devices fit well into such situations. However, we also think that IPADs also have the potential to become useful in many current workplaces, especially since these often permit a high degree of local "micro-mobility" that means that alternatives to desktop-based solutions will be needed. We believe that Hummingbirds and other IPADs may prove to be a useful tool for supporting group collaboration in the future.

8 Acknowledgements

Thanks to the people who participated in the user testing of the Hummingbirds, including Charlotte Averman, Staffan Björk, Roberto Busso, Johan Redström, Ella Tallyn and Karl-Petter Åkesson, and to our "user model", Erica Wollerfjord. Thanks to Henrik Fagrell, Fredrik Ljungberg, Peter Ljungstrand, Johan Redström and many other colleagues at the Viktoria Institute for valuable comments. Thanks also to Jona J. Bjur and Gunilla Grahn for thoughts about the design of future IPADs, and to Thomas Gabinus and others at Ericsson Microwave for early encouragement. Finally, thanks to Hans W. Gellersen for arranging the *Workshop on Mobile CSCW* where this work was first presented, to the workshop participants for their feedback, and to the reviewers whose suggestions helped to greatly improve this paper. This work was part of the Mobile Media project in the Mobile Informatics research programme, funded by SITI, the Swedish Research Institute for Information Technology. Additional funding was provided by the Intelligent Environments project in the NUTEK PROMODIS research program.

9 References

1. Beadle, H.W.P., Maguire Jr., G.Q., Smith, M.T. Location Based Personal Mobile Computing and Communication. In *Proc. 9th IEEE Workshop on Local and Metropolitan Area Networks*, IEEE Press, 1998.
2. Belotti, V. and Bly, S. Walking Away from the Desktop Computer: Distributed Collaboration and Mobility in a Product Design Team. In *Proceedings of CSCW '96,* pp 209-218, ACM Press, New York, 1996.
3. Borovoy, R., Martin, F. Resnick, M. and Silverman, B. GroupWear: Nametags that Tell About Relationships. In *CHI '98 Summary,* pp. 329-330, ACM Press, New York, 1998.
4. Borovoy, R., Martin, F., Vemuri, S., Resnick, M., Silverman, B. and Hancock, C. Meme Tags and Community Mirrors: Moving from Conferences to Collaboration. In *Proceedings of CSCW '98,* pp. 159-168, ACM Press, New York, 1998.
5. Brave, S., Ishii, H., and Dahley, A. Tangible Interfaces for Remote Collaboration and Communication. In *Proceedings of CSCW '98,* pp. 169-178, ACM Press, New York, 1998.
6. Dourish, P. and Bly, S. Portholes: Supporting Awareness in a Distributed Work Group. In *Proceedings of CHI '92,* pp. 541-547, ACM Press, New York, 1992.
7. Fogg, B.J., Cutler, L.D., Arnold, P. and Eisbach, C. HandJive: A Device for Interpersonal Entertainment. In *Proceedings of CHI '98,* pp. 57-64, ACM Press, New York, 1998.
8. Greenberg, S. Peepholes: Low Cost Awareness of One's Community. In *CHI '96 Companion*, pp. 206-207, ACM Press, New York, 1996.
9. Iwatani, Y. Love: Japanese Style. *Wired News* (Web-based news service), 11 June, 1998. URL: http://www.wired.com/news/culture/story/12899.html
10. Tollmar, K., Sandor, O and Shömer, A. Supporting Social Awareness @Work, Design and Experience. In *Proceedings of CSCW '96,* pp. 298-307, ACM Press, New York, 1996.
11. Want, R., Hopper, A., Falcao, V. and Gibbons, J. The Active Badge Location System. *ACM Transactions on Information Systems,* Vol. 10 (1), 1992.
12. Weiser. M. and Brown, J. Designing Calm Technology. *Powergrid Journal* 1.01 (Web journal), 1996. URL: http://www.powergrid.com/1.01/calmtech.html

13. Whittaker, S., Frohlich, D. and Daly-Jones, O. Informal Workplace Communication: What Is It Like and How Can We Support It? In *Proceedings of CHI '94*, pp. 208-215, ACM Press, New York, 1994.

Token-Based Access to Digital Information

Lars Erik Holmquist, Johan Redström and Peter Ljungstrand

Abstract. Several systems have been designed where a physical object is used to access digital information that is stored outside the object, but as yet no common vocabulary exists to describe such systems. We introduce a schema with three types of physical objects that can be linked to digital information: *Containers* are generic objects used to move information between different devices or platforms; *tokens* are used to access stored information, the nature of which is physically reflected in the token in some way; and *tools* are used to manipulate digital information. This paper gives special notice to token-based access system, and design implications for such systems are discussed. As an example of token-based access we have implemented *WebStickers*, where physical objects can be coupled with WWW pages. We present some examples of how tokens are used to access digital information in this system, and discuss future work in this area.

1 Introduction

In recent years, one of the most compelling visions of the future of computers has been that of *ubiquitous computing*, where computers would leave the desktop and move into the world that surrounds us [14]. By shifting the emphasis from the universal functionality of desktop workstations to small, dedicated computational tools, proposed ubiquitous computing environments hope to make computers as readily available and easy to use as notepads and whiteboards. In some ways, this vision is starting to make its way to reality, and with the continued miniaturisation and decreasing prices of PDAs and embedded processors, much of the technology required to make these visions a reality now exists.

However, with the increased power and complexity of portable computers, there is also the risk of simply replacing one problem with another. By moving all computing functions from one platform to another, perhaps we will not always gain as much as we would hope. Even worse, advantages taken for granted with stationary computers (large screens, high computational power, access to high-speed networks, etc.) are often missing on mobile devices. There is a risk that rather than simplifying the use of computers, the proliferation of a multitude of computational devices will instead make for higher complexity – thus achieving the opposite of the goal of ubiquitous computing.

An alternative approach to accessing and manipulating digital information is to use physical objects that are not in themselves computers, but nevertheless are used for representing information. Most types of information today exists in digital form, including text, images, music, films, and so on. With a suitable infrastructure, it should be feasible to have access to any book ever written, every piece of music ever recorded, any piece of art ever painted, anytime, anywhere, without the need for a physical carrier. However, this might also lead to serious problems in designing the human interface; experiences with the World Wide Web have already shown us that designing the interface to a practically limitless information space is extremely difficult.

But humans are inherently good at managing physical space, by ordering and sorting artifacts in their environment. Our senses give us many clues to the properties of physical objects, so that we are able to draw many useful conclusions from the way objects look and feel and how they are arranged in our environment [3]. We might take advantage of some of these capabilities when designing systems for accessing digital information, by using physical representations that are in themselves not carriers of information, but act as pointers to some online data.

In this paper, we will examine several such systems, concentrating on approaches where digital information is distributed using physical objects that represent some digital information or computational function. The process of accessing virtual data through a physical object we will term *token-based access to digital information.* The purpose of this paper is to systematise the properties of such systems, and to put them in relation to systems using other approaches, thus forming the basis for a discussion of how we can use properties in the physical world to help us better interact with distributed digital information.

2 Physical Objects as Representations of Information

There is a long history of the use of physical items to represent information, without the item actually containing the information that it represents (cf. [6, 16]). Souvenirs, photographs and keepsakes aid in the remembrance of places, past events and persons, by acting as a trigger for the user to remember certain information. The pieces used in board-games act as representations of the players through which they can perform their actions (cf. [15]). Gambling tokens used in casinos represents a value that is not inherent in the actual piece of plastic, much like the value of paper money traditionally has been guaranteed by a government's gold reserve. Cards of various kinds (calling cards, debit cards, etc.) are used to access assets – telephone call minutes, money stored in a bank account, etc. – that are not stored in the physical cards themselves.

Similarly, tokens in human-computer interaction will trigger the display of information that is digitally stored outside the token in some way. In the research community, several recent systems use physical objects without any inherent computational properties as representations of digital information in some way or another, but there is as yet a lack of vocabulary for describing and analyzing such systems. To facilitate a discussion, we will first introduce three different classes of physical objects that represent digital information or computational functions: *tokens*, *tools* and *containers*.

2.1 Tokens, Tools and Containers

We will call an object a *container*, if it is a generic object that can be associated with any type of digital information. We will call it a *token*, if the digital information associated with the object is reflected in the physical properties of the token in some way, thus making the object more closely tied to the information it represents. Finally, some physical objects are to be considered as *tools*, since they are used to actively manipulate digital information, usually by representing some kind of computational function. Some accounts of related work should help clarifying these distinctions.

Containers. Several systems have been proposed in which digital information can be attached to physical objects, often to simplify the task of moving information between various computers and/or display devices. In the *pick-and-drop* approach [7], a pen was used as a container to physically "pick-and-drop" digital information between computers, analogous to how icons are “dragged-and-dropped” on a single screen. *Informative Things* [1] let ordinary floppy disks act as pointers to on-line information by associating them with a digital ID. A disk could thus be shared between users as usual, but would seem to have “endless” storage, since no information apart from the ID was actually stored on the disk. The authors also discuss some future scenarios where other objects might be used as “Things”. *mediaBlocks* were small wooden blocks which let digital information be stored and accessed through a variety of different means [12]; for instance, after first associating a block to a digital whiteboard, the block could be used to transfer the scribbles on the whiteboard to a laser printer for printout. Finally, in the *Passage* system [8] information of various kinds could be moved between different computers by “attaching” it to small physical objects called “passengers”.

Although all these systems in some sense could be said to use “tokens” to represent digital information, we prefer to call these objects *containers*. Unlike what we will term tokens, containers are generic, in that the physical properties of a container do not reflect the nature of the digital information it is associated with. Taking mediaBlocks as an example, note that by merely examining the physical form it is impossible to know if a block is associated with say a video clip, a PowerPoint presentation or a whiteboard scribble. This generic quality makes containers potentially very useful for the distribution and manipulation of a variety of digital information, but it also means that containers do not provide any additional cognitive cues for the user as to what their “contents” are. Furthermore, containers are mostly used for short-term distribution and access, making them inherently transient in nature.

Tokens. In our definition, *tokens* are objects that physically resemble the information they represent in some way. Tokens are typically only transient if the token itself is short-lived. In the *metaDESK* map-display system [11], a set of objects were designed to physically resemble different buildings appearing on a digital map. By placing the models on a horizontal display, users could bring up the relevant portion of the map, and the physical form of the objects would serve as a cognitive aid for the user in finding the right part of the map to display. In the *ambient-ROOM* [4], objects were used to represent various types of information, and by bringing an object to an information display, an “ambient” display of that information could be accessed. For instance, by bringing a toy car close to a speaker, ambient sounds reflecting the activities in a toy project could be heard.

In the electronic tagging system described in [13], an object could be augmented with a digital ID tag allowing it to be linked to some digital information, thus letting the physical objects act as a pointer to the digital information. Some examples included a book that was associated with appropriate electronic information, such as the author's web page or a relevant page at an on-line bookstore, and a watch that was associated with the user's on-line calendar. Similarly, in the *WebStickers* system [5], users could attach barcode stickers to objects, and then associate a barcode to a web page that was somehow relevant to the object. (This system will be described in more detail later.)

Tools. Finally, some physical objects are used as representations of computational functions. We will call such objects *tools*. Some tools act as "handles" to manipulate virtual objects. In the *Bricks* system [2], a physical "brick" was attached to a graphical object on a horizontal display, and could then be used to move and rotate the on-screen object. By employing two bricks, a graphical object could be scaled and distorted. Some tools physically resemble the computational function they represent. In the metaDESK system, a physical representations of a magnifying glass was used to invoke functions similar to those of the *magic lenses* explored in graphical UIs [9]. By manipulating the physical magnifying glass, the user could apply the lens functions to a part of the map, thus seeing an alternative display "through" the lens represented by the magnifying glass. Other physical representations such as a "flashlight" were also used. In the electronic tagging system mentioned above (cf. [13]), a French dictionary was associated with a language translation function, so that a text could be translated simply by bringing the physical representation close to the screen where the text was displayed.

Sometimes the distinction between a tool and a token or a container will blur, since when a physical object is attached to a virtual, direct manipulation of virtual properties using the physical representation might become possible. In the metaDESK, models of buildings (tokens) were also used to scale and rotate a map, analogous to the Bricks system. In mediaBlocks, several mediaBlocks (containers) could be used in conjunctions with a workbench to sequence a presentation; the completed presentation could then be associated with a new block. Such "hybrid" systems, where a physical representation has several possible uses depending on the context, are an area where we expect to see much development, but we will consider them outside the scope of this paper.

2.2 A Note on Vocabulary

The definition of *token* in the online edition of the Merriam-Webster Collegiate Dictionary includes:

> **1** : an outward sign or expression <his tears were tokens of his grief>
> **2 a** : SYMBOL, EMBLEM <a white flag is a token of surrender> **b** : an instance of a linguistic expression
> **3** : a distinguishing feature : CHARACTERISTIC
> **4 a** : SOUVENIR, KEEPSAKE **b** : a small part representing the whole : INDICATION <this is only a token of what we hope to accomplish> **c** : something given or shown as a guarantee (as of authority, right, or identity)
> (Note: meanings **5** – *resembling money* – and **6** – *tokenism* – have been excluded)

Our intention with this choice of word is to show that a token is a "small part representing the whole", in that properties of the digital information are reflected in the token, and that the token should have some characteristic of the information it is linked to. We considered using some other term, in particular the word *phicon*, which has been used for physical counterparts to GUI icons, but decided against it. In the literature, the term phicon has been used both for what we define as tokens (e.g. the models of buildings in the metaDESK [11]) and for containers (e.g. mediaBlocks [12]), creating some confusion, which we sought to avoid with this choice of terms.

3 Token-Based Access to Digital Information

As we have seen, there are several different approaches to how we can let a physical object represent some kind of digital data or computational function. We will in the following concentrate on what we term *token-based access to digital information,* because this is an area that provides many design opportunities that should be further explored. We will define token-based access to digital information as:

> *A system where a physical object (token) is used to access some digital information that is stored outside the object, and where the physical representation in some way reflects the nature of the digital information it is associated with*

A token is a representation of some digital information, but only by association and resemblance – a token is not a computer or a display. Instead, the user will have to bring the token to some kind of external device to access the associated information.

3.1 Components

In a token-based interaction system, users will need to have access to two types of components:

- *A set of physical objects which are used as representation of some digital information.* These objects we will call tokens
- *A set of access points for the digital information associated with tokens.* These access points we will call *information faucets*, or faucets for short

We have chosen the term *faucet* rather than a term such as *display*, since it can be any type of device capable of presenting information, not just a graphical computer display – perhaps a speaker, a tactile device, etc. Importantly, while a token is by definition not a computer (it typically contains no computational power), neither should a faucet be considered as a computer from the user's point of view. Instead, tokens and faucets together comprise a system that provides users access to digital information – the fact that computer technology, networks, etc., might feature heavily in the implementation of such a system should not need to be of concern to the user.

3.2 Interaction in token-based access systems

Interacting with tokens can be either to access the information associated with a certain token, or to create or modify such associations. These two aspects of the human-computer interaction we will call *access* and *association* respectively.

Access. Fundamental for any token-based system is that it allows a user to access a certain piece of information by means of presenting a token to an information faucet. By controlling the availability of tokens it is possible to control the access to information. For instance, if we allow for a number of copies of the same token to be made, several people will be able to access the information, perhaps simultaneously. Conversely, if we want to restrict access, we might only allow one instance of a token to be produced, and through some measures make it impossible to copy, thus letting the token act as the single “key” to the information in question.

We might also want to introduce some additional constraints on information access. For instance, a *combination* of tokens might be used to access the information associated with all the tokens simultaneously. A more interesting option is to use the combinations as such to form criteria for information access. For example, if two tokens represent work in a joint project, certain aspects of that work might only be accessible when both tokens are presented simultaneously, much like we might require more than one key to open a door.

Depending on the present purpose, information access might be constrained by physical *location* as well. For example, some information might only be applicable at a given location (e.g. a building) and by using tokens that only work with local faucets any distribution beyond that location can be limited. Correspondingly, public information that is meant to be widely distributed will have to use tokens that do not pose such a limitation, but instead are applicable to variety of faucets.

Association. If the association of digital information with a physical token is unconstrained and at any time allows the user to re-associate the token with any other piece of information, we are close to the properties of containers. However, when using tokens it is more interesting to investigate different ways of constraining the set of possible associations. For example, we might want to restrict the associations of a certain kind of tokens to a certain kind of information, thus avoiding some confusions between the how the properties of the token are reflected in the information it represents. We might also make the associations fixed once and for all, making the connection between the token and a certain piece of information as static as possible. This would typically be the case in a public display system, say an interactive museum exhibit, where one would not want the users to be able to change the way information is associated with the physical objects on display.

Further, we might allow a user to associate more than one piece of information to a certain token. This we may call *overloading*. Overloading a token with information might have various effects. For instance, the token might represent different pieces of information at different locations or in different contexts, as is often the case with everyday objects. Alternatively, the user might be able to access several different pieces of information at the same time when applying the token to a faucet. In the latter case, the information might be displayed with a choice of which information to present.

4 A Sample System for Token-Based Access: WebStickers

As an example of token-based interaction, we have developed the WebStickers system [5]. This system is quite flexible, in that it uses the Internet for distribution of data, and thus we can use any computer with the appropriate (off-the-shelf) hardware as a faucet. The system allows users to couple identifiers in the form of barcodes to locations on the World Wide Web. Users are given a set of stickers with pre-printed unique barcodes, and can then attach the stickers to any object they want to use as bookmark. Users then use a barcode scanner to associate a barcode with one or more web pages, and are able to return to a page by again scanning the corresponding barcode. The idea is to allow users to take advantage of the properties of any object in their surroundings and use these properties as cognitive cues to the finding a certain web location.

The system is implemented as a database accessible via HTTP. In the database, identifiers in the form of unique character strings are coupled with URLs. An off-the-shelf barcode reader is used to scan barcode stickers, which are printed on sheets of adhesive stickers using a standard laser printer. A small client application on the user's computer monitors incoming characters from the barcode reader, matching identifiers with URLs by calling the on-line database, and displaying the corresponding web page in the user's browser. To create new associations, the user simply change the mode of the client program from *Goto* to *Learn*, and the currently displayed web page is associated with the scanned barcode in the server database. Using codes coupled with URLs in a database, rather than coding URLs directly into barcodes, makes it possible to create new associations or change old associations easily.

4.1 Modes of Interaction

The WebStickers system provides a basic form of access to web pages through tokens. There is currently no provision for more advanced access forms, such as those provided by combinations of tokens or based on specific locations. As for association, WebStickers currently allows totally free association between web pages and tokens, placing the responsibility of finding the correct token on the user making the association. This is reasonable considering the experimental nature of the current system, but in future versions it might be useful to introduce some restrictions. Introducing ready-made tokens for specific tasks might also be considered. (We already have one such ready-made token in the form of Post-It notes – see below.) WebStickers does allow for a form of overloading, by letting the user associate more than one web page with a single token. When such a token is accessed, the user is presented with an intermediate web page where she can choose from a list of URLs.

4.2 Types of Tokens

With WebStickers, we have been able to experiment with a variety of different tokens as representations of web-based information. Here are some examples.

Transient tokens. For web page bookmarks that are only meant to be kept for a short time, say no more than a few weeks, we have been using books of Post-It notes with pre-printed barcodes. After associating a note with a web page users can then scribble a comment on the note that helps them remember what web page the note refers to, and attach it to their screen, their notice board, someone else's door, etc. Post-Its are explicitly designed for short-term information, making them ideal tokens to represent transient web bookmarks. After a while the glue in the note will cease functioning and the note will fall off whatever surface it is attached to, at which time the user can select to transfer the bookmark to a more permanent location, or discard it completely.

Tokens with a direct digital analogy. Some WWW bookmarks have a direct counterpart in the real world. For instance, when referring to the proceedings from a conference, it is often more comfortable to use the physical book than to read from an on-line proceedings page. However, when a paper is to be e-mailed to someone else, when it is to be searched for specific terms, when we need to quote some sections, etc., having easy access to the electronic version is useful. We have been using the pre-existing barcodes on conference proceedings for coupling them to their on-line counterpart. Since a book of proceedings is an archival object, it will mostly be stored away on a bookshelf. When working with a book, the user will take it down and bring it to her desk, and now through the WebStickers association she can have immediate access to the corresponding on-line documents as well.

Tokens tied to a certain activity. We have experimented with using objects that are tied to a specific activity as bookmarks to related web pages. A Swedish-English dictionary has been associated with the web page of the Encyclopaedia Britannica, the thought being that when users are searching for a word this web page will come in handy if the physical dictionary is not sufficient. Similarly, a user has tied the cup used for drinking the morning coffee to the URL of the morning news (made available on the Internet by the national radio station), thus tying the activity of drinking coffee to listening to news updates.

4.3 Conclusions from the WebStickers system

By constructing a system for token-based access that allows a wide variety of tokens to be associated with a very large information space (the World Wide Web), we have been able gain experience in how virtual properties can be reflected in physical objects. We have found some very obvious correspondences, such as that between Post-It-notes and transient bookmarks, but feel that it would be useful to generalize the discussion of how to design tokens. In the following, some initial design ideas for future token-based systems will be given.

5 Fitting the Token to the Task

Since a token typically will need to have little or no inherent computational resources, many of the constraints posed on the design of ordinary computers will not have much effect. For example, a token will not need any display; it will not need to have a processor or a power supply; it will be much less sensitive to wear and tear, and so on. This leaves us with far more freedom to design and build the tokens according to other criteria.

The most important criterion will be to design the tokens in a way that clearly displays what they represent and what can be done with them, i.e. their *affordances* [3]. Matching the affordances of the token with the task it is designed to be used in, can be done in a number of different ways including the use of different materials, sizes and shapes. Since tokens are not self-contained but tied to information faucets, the interaction can also be designed to take other factors into consideration, such as the physical position or usage context.

Just like when designing graphical interfaces, care must be taken when designing tokens. For instance, often certain shapes or colors convey values or meaning specific to a culture, like the symbol of the cross does in Christian religions. Whether such cultural values should be used or avoided, will depend on the kind of information to represent and who are going to use it in what context. However, token-based interaction systems will be less loaded with predefined meaning if strongly established symbols are avoided. Below we will sketch some of the possibilities for how the properties of digital information can be reflected in the design of token-based interaction.

5.1 Materials

Tokens can be made in a variety of materials depending on what they should represent. Tokens that represent information that is only meant to last for a short while might be made of material that wears out easily. Consider for example the difference in paper quality between books and newspapers, and in the glue used on Post-It notes and postage stamps. Here, the lack of durability of the newspaper and the glue on the Post-It note are not faulty but intended, since they represent information which is only intended to be used for a short time. A book, on the other hand, is intended to be kept for some time, and a stamp should stay stuck on the envelope that it was attached to.

Similarly, tokens made in fragile materials can be used to represent information that should be handled with care. Tokens made in very heavy materials can be used to represent information that is not supposed to be transported very far from its current location. Tokens representing information that is to be used frequently by a certain user might take the form of jewelry or perhaps a belt made in some comfortable material.

5.2 Sizes and Shapes

Tokens can come in many different sizes and shapes depending on the purpose. For example: tokens that are meant to be passed between users should be graspable. Tokens that are private should afford hiding and must thus be small enough to fit into a pocket or perhaps into the palm of a closed hand. Very large tokens will be harder to move without attracting attention and thus suitable to represent information that is of public interest. If we have a large number of tokens that we need to store in the same place we might want to make them easy to stack or pile. They will then have to have a size and shape that afford this, meaning that tokens similar to cards or discs might be more suitable than tokens similar to marbles.

Further, the size and shape of the tokens can help restricting their use to avoid mistakes. Consider puzzles: besides the color of a piece, its shape determines where in the puzzle the piece can be applied. This is especially obvious in puzzles made for small children where each of the very few pieces fit into a certain slot. In the case of token-based interaction, using shapes that only fit in certain slots can be used to determine which information faucets are applicable. If the information the token represents is of a kind that only can be accessed in certain information faucets, the shape of the token can be made in a way that only will fit into proper kind of faucets.

5.3 Usage Context

Everyday objects are often used within a special context, and when moved out of that context their "meaning" tend to change. As an example, take the many knives used in a kitchen for different purposes, e.g. cutting bread, meat, fish etc. Sometimes they are stored in drawers in the kitchen. Now imagine what happens if we instead store them in another drawer in the apartment, say, where you usually store your socks or underwear. If someone found your kitchen knifes in your bedroom drawer, he or she would definitely react differently compared to if he or she had found them in the kitchen. Thus, the very location of tools and objects can convey meaning. This should be acknowledged when using tokens for interacting with computers, by means of for example how to constrain access to (e.g. *location* and *combinations* of tokens) and associations (*overloading* tokens) with information.

Thus, we have seen how a wide variety of virtual or digital properties can be reflected in the design of the components of a token-based access system. We have in this paper only been able to sketch the outlines of these possibilities, and many practical design experiments and evaluations will be needed before any firm conclusions can be drawn or any solid design specifications can be given.

6 Conclusions and Future Work

We have attempted to show that token-based access to digital information is a valid interaction paradigm that can be used to support access to information in a distributed computing environment. Token-based access systems differ from container-based systems in that they imply a stronger coupling between physical and virtual data, i.e. the properties of a token should reflect the properties of the data it is associated with. This makes it possible to design tokens that provide users with a strong cognitive support for accessing information in distributed systems. It also opens many possibilities for building in aspects of the user interaction into the token itself, rather than having these solely confined to the virtual domain. For instance, by designing tokens with certain physical properties, say tokens that are easy or difficult to share between users, it is possible to have some desired affordances physically reflected in the token.

For future commercial applications, we can see many situations where it would be more convenient to use token-based access than a physical carrier of information. The music business is currently a good example. With forays already being made into distributing music on the Internet using the MP3 format, in the future it might be feasible that rather than buying a music carrier such as a CD or DVD, consumers will purchase a small token representing a recording. By bringing such a token to a suitable player (faucet), the user can then listen to the music associated with the token. Unlike a CD, the token would never run the risk of being scratched, and through encryption of unique IDs on each token, music companies can make sure that their music is protected. Technical realization of such a system is already possible [10].

As we have seen, several systems for token-based access to digital information have already been realized in research labs, and several systems have also been constructed where physical containers and tools are used to distribute and manipulate data. This serves to prove the technical validity of such systems, and technology for tagging and sensing objects is already good enough to construct useful applications. However, neither this paper nor most previous work has been able to more than touch on some of the most important aspects of token-based access.

In particular, matters concerning security, privacy and rights concerning information associated with tokens need to be considered. Can valuable information be safely made available on public networks without the risk for unauthorized access? Should tokens be possible to copy, and what will then happen to the information and associated access rights? Who should have the right to modify materials associated with a token, and who should be allowed to modify the associations themselves? In the experimental applications, the impact of such decisions has been limited, since the systems have been used only to a limited extent and by a limited audience of mainly expert users, but in the future these questions may come to have a serious impact. The validity of token-based access to digital information is probably more dependent on the resolution of such issues than any technological hurdles.

Before general token-based systems break into the mainstream, we will have to take these matters into consideration, and will also have to refine the way such systems are designed, improving their properties from a user perspective. In this paper we have sketched some initial possibilities for tokens-based access to digital information, but much more work needs to be done in this area. This work must be guided by experiences in disciplines such as user-interface design, industrial design and ergonomics, making for a truly cross-disciplinary challenge. We believe that with the correct approach, systems offering token-based access to digital information can prove very useful in the development of future distributed computing environments.

7 Acknowledgements

Thanks to colleagues at the Viktoria Institute and to the HUC ’99 reviewers, whose comments helped to improve this paper. This work was performed in the project *Intelligent Environments*, part of the PROMODIS research program funded by NUTEK, the Swedish National Board for Industrial and Technical Development. Additional funding came from the *Mobile Informatics* program sponsored by SITI, the Swedish Institute for Information Technology.

8 References

1. Barrett, R. and Maglio, P.P. Informative Things: How to attach information to the real world. In *Proceedings of UIST '98,* ACM Press, 1998.
2. Fitzmaurice, G.W., Ishii, H and Buxton, W. Bricks: Laying the Foundations for Graspable User Interfaces. In *Proceedings of CHI '95,* pp. 442-449 ACM Press, 1995.
3. Gibson, J. J. *The Ecological Approach to Visual Perception.* Lawrence Erlbaum Assoc., 1979.
4. Ishii, H. and Ullmer, B. Tangible Bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of CHI '97,* ACM Press, 1997.
5. Ljungstrand, P. and Holmquist, L.E. WebStickers: Using Physical Objects as WWW Bookmarks. In *Extended Abstracts of CHI '99,* ACM Press, 1999.
6. Norman, D. A. *Thing That Make Us Smart*. Perseus Books, 1993.
7. Rekimoto, J. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of UIST '97,* ACM Press, 1997.
8. Streitz, N.A., Geissler, J., Holmer, T. et al. I-LAND: An Interactive Landscape for Creativity and Innovation. In *Proceedings of CHI '99,* ACM Press, 1999.
9. Stone, M., Fishkin, K. and Bier, E. The Movable Filter as a User Interface Tool. In *Proceedings of CHI '94,* ACM Press, pp. 306-312, 1994.
10. Talbot, C. *Honey I Shrunk the CD.* http://www.media.mit.edu/pia/Research/CDs/
11. Ullmer, B. and Ishii, H. The metaDESK: Models and Prototypes for Tangible User Interfaces. In *Proceedings of UIST '97,* ACM Press, 1997.
12. Ullmer, B., Ishii, H. and Glas, D. mediaBlocks: Physical Containers, Transports, and Controls for Online Media. In *Proceedings of SIGGRAPH '98,* ACM Press,1998.
13. Want, R., Fishkin, K.P., Gujar, A. and Harrison, B.L. Bridging Physical and Virtual Worlds with Electronic Tags. In *Proceedings of CHI '99,* ACM Press, 1999.
14. Weiser, M. The Computer for the 21st Century. *Scientific American,* 265 (3), pp. 94-104, 1991.
15. Zhang, J. The Interaction between Perceptual and Cognitive Processes in a Distributed Problem Solving Task. In *Working Notes of*

the 1993 AAAI Fall Symposium on Games: Planning and Learning, 1993.

16. Zhang, J. and Norman, D. A. Representations in Distributed Cognitive Tasks. *Cognitive Science*, vol. 18, pp. 87-122, 1994.

Gothenburg Studies in Informatics
Department of Informatics
Göteborg University
Viktoriagatan 13. Box 620
S- 405 30 Göteborg, Sweden
E-mail: reports@informatics.gu.se
WWW: www.informatics.gu.se