

## SUPPORTING DSP EDUCATION USING JAVA

*Dr. Mike Jackson, Dr. David I. Laurenson, and Prof. Bernie Mulgrew*

Department of Electronics and Electrical Engineering,  
The University of Edinburgh,  
Kings Buildings, Mayfield Road,  
Edinburgh,  
United Kingdom  
EH9 3JL

mjj@ee.ed.ac.uk dil@ee.ed.ac.uk bernie@ee.ed.ac.uk

### ABSTRACT

This paper describes the authors' research into using Java to develop tools to support the teaching of Digital Signal Processing. Focusing on the application of concepts of Human-Computer Interaction allows the formulation of design requirements which facilitate the development of tools that are both usable and useful. Developing tools meeting these requirements yields teaching aids endowed with a novel characteristic lacking from comparable Java tools — flexibility. An exemplar of this flexibility is that educators with no Java experience can configure the tools to meet course requirements. A planned user-centred evaluation is intended to provide information that lays the groundwork for investigating possible educational gains resulting from the use of these tools by educators and students.

### 1. INTRODUCTION

When considering the development of CAL (Computer-Assisted Learning) tools to support the teaching of Digital Signal Processing (DSP) choosing to use a commercial environment, for example MathWorks MATLAB, can be tempting. Not only is there support for the construction of simulations but there is the ability to visualise these dynamically within a graphical user-interface, potentially yielding a straightforward-to-use teaching tool. Furthermore, the users of such an educational tool — students — are provided with experience of using an environment they could encounter in their future academic and engineering careers.

However, adopting a commercial environment requires that students be taught to use the environment before being able to access and use the educational tools running under that environment. This could be time-consuming and can conflict with the aims of academic institutions whose primary concern is educating students in the fundamental concepts of a field not in the use of commercial software.

---

This work was supported by a SHEFC (Scottish Higher Education Funding Council) research grant held jointly with the Department of Electronics and Electrical Engineering, University of Strathclyde, Glasgow, United Kingdom.

In addition, using the, typically licenced, environments requires students to be on University premises or to have access to University systems. Typically, this access will only be authorised within certain hours, not necessarily coincident with a student's preferred study times. While the use of publicly-available environments such as Octave [1] could mitigate these locational restrictions they incur the necessity of downloading and installing the environment on systems where they are not already available. In striving to use a tool to learn about DSP a student may be required to undertake a range of unrelated, and potentially time-consuming, activities.

One solution to this problem is to exploit the technology of the World-Wide Web. In particular, using Java as a development environment allows the provision of educational tools to students and other interested parties in a more flexible and accessible way than is achievable using proprietary development environments such as MATLAB. Tools can be accessed by students wherever and whenever they wish — conditional only on an Internet connection and compatible browser — or even off-line, via disk, providing less locational and temporal constraints on their study behaviour, removing one source of work-related pressure from students. As Picone *et al.* [2] highlight, the need to download esoteric tools is removed since leading browsers provide support for Java. Furthermore, the only technical expertise required from students is that they be able to operate a web browser — a not unreasonable expectation in today's world.

Investigating the provision of Java tools to support DSP education is one concern of the WebEng Project (Web-based Procedures, Tools and Strategies for Internet-based Engineering Education) <sup>1</sup>. The WebEng project is being undertaken jointly by the Departments of Electronics and Electrical Engineering at the Universities of Strathclyde and Edinburgh and focuses on the development of a unified, on-line, web-based learning and teaching environment for DSP education. Other projects are also focusing on similar unified environments incorporating on-line teaching resources and tools, for example the Johns Hopkins Signals, Systems and Control Project [3] and the Systool project [4].

---

<sup>1</sup>The project web-site is at <http://www.webeng.org>

This paper discusses the issues that arose during the development of a suite of Java tools for DSP education by the Edinburgh members of the WebEng project. In §2 principles of Human–Computer Interaction are explored to motivate design requirements for the tools. In §3 the key characteristics of the tools developed are then described. It is argued that the authors’ tools are unique — in comparison to similar web–based tools — in terms of the flexibility of use supported for both students and educators. In §4 a user–centred evaluation, intended to provide information to guide further development of the tools and to lay the groundwork for investigating issues relating to educational gains resulting from the use of these tools, is described. Finally, section §5 summarises the work described in this paper.

## 2. ISSUES AND REQUIREMENTS

The area of Human–Computer Interaction (HCI) is concerned with the construction of systems that are both usable — allow users to accomplish their tasks in a straightforward manner — and useful — allow users to accomplish tasks meaningful to them. A cornerstone of HCI is that an understanding of four key areas is vital if the design of usable and useful systems is to be effected. The four areas are [5]:

- the intended **users** of the computer–based system;
- the **tasks** the computer–based system is intended to help the users to undertake;
- the **technology** available to support the users in the execution of their tasks;
- the **environment** in which the users perform their tasks.

Exploring and analysing these four key areas creates requirements that can guide the development of tools that effectively support a user in the performance of their task. In the following sections these issues are examined in more detail with respect to the tool development undertaken by the authors.

### 2.1. Users

The target audiences for the tools are both educators (in this context lecturers or tutors) and students. For educators the tools serve as a teaching aid to support the explanation of certain concepts to students in tutorials or lectures. One requirement that immediately arises is that the tools must provide features that traditional teaching aids, including books, overhead slides or white–boards, do not. Failure to address this requirement results in the development of tools which provide no incentive for their uptake by educators.

An advantage of using computer–based tools, over books and slides, in lectures and tutorials, as well as in student study activity, is that dynamic visualisation can be exploited. It is intuitively more effective — and easier — for an educator to present a single animation, under their control, than a series of static diagrams. It is therefore an important development guideline that animation be exploited wherever appropriate to re–enforce the teaching of certain

concepts in a way not possible using traditional teaching materials.

With respect to students the tools act as study aids, facilitating within them an improved understanding of DSP concepts. The knowledge that latter–year undergraduate students — the prime intended student audience — can have conflicting demands on their time from a variety of courses, in terms of study and course–work, generates the requirement that any tools should be intuitive to use. The time a student spends with a tool should be consumed by focusing on the DSP concepts the tool is intended to impart and not on learning how to use the tool. This is an important distinction — people do not learn to use tools for the sake of it, rather as a stepping stone to using the tool to achieve other, more relevant or interesting, goals. This requirement is re–enforced given that users of software systems are not inclined to take the time to read documentation or perform familiarisation exercises but rather to plunge in and get ”hands–on” experience using the tool to accomplish their own tasks [6].

### 2.2. Task

Educators and students have contrasting yet related, tasks to perform. On the one hand educators are undertaking the task of teaching students DSP concepts and on the other students are learning about DSP concepts and striving to understand these.

To facilitate the teaching of DSP it is vital that the tools support the teaching of concepts in a way desirable to educators and can be integrated into their courses. Different educators may, for example, use different examples to highlight and explain the same concepts. It is vital therefore that any tools can conform to an educator’s requirements and be integrated into their tried–and–tested course structure rather than educators having to change aspects of their courses to fit with what the tools provide. This creates the requirement that the tools should allow educators to have control over their content.

One way in which tools can be developed to support the students’ contrasting aim of learning and understanding is to support learning by observation. While ”movies” such as Watson’s rotating phasor animation [7] can be useful, a more enhanced learning experience could be facilitated by allowing the users to have more control over the tools — moving towards learning by experimentation or doing. Supporting student manipulation of key parameters or allowing them to provide their own signals and data — as supported by the convolution tool described in [2] — facilitates experimentation with textbook or lecture examples and modified versions of these. Crutchfield and Rugh [8] explain how such interaction can allow students to sketch out and explore ideas for course–work solutions before setting down to do the mathematics required by the course–work. The fact that interaction can facilitate the more effective integration of such tools into a course by supporting the completion of course–work or forming part of a tutorial exercise — by which students are set some problem to investigate and must use the tool as part of the investigative process — could prove useful to counter negative experiences regarding the usage patterns of such tools by students recounted

in §4.4.

### 2.3. Technology

There are a wide range of platforms and software under which Java tools can be run. Besides from the prevalent hardware platforms — PC, UNIX workstation and Macintosh — there are also two prevalent Internet browsers — Microsoft Internet Explorer and Netscape. These browsers also come in a myriad of versions varying in their ability to support the versions of Java that are currently in existence — some providing immediate support to some Java flavours and others requiring the presence of plug-ins. This is a consequence of browser support for Java lagging behind the development of the Java language.

Such variability in platform and browser technology motivates the requirement that the tools be operable on as wide a range of platforms as possible to allow their use by as great an interested audience as possible. This facilitates flexibility of access, not unnecessarily constraining students as to where and when they can use the tools, thereby avoiding one of the problems inherent in the use of environments like MATLAB. In addition, the version of Java should be selected so that the necessity for students to download additional plug-ins or software is avoided as much as possible, allowing them to focus on learning DSP and not the unrelated activities of downloading and installing software.

Another technology-imposed requirement concerns display size. Owners of older hardware may be restricted to using small 640 x 480 or 800 x 600 size monitors. Related to this is the fact that browsers can further restrict the available display space with their menu-bars, icons and other interface features. For applets that are embedded in web-pages this can result in the user being unable to view the entire applet interface within their display, a problem exhibited by the tools of [4] and [3]. So as not to deny users of small displays the ability to satisfactorily use the tools creates the requirement that the tools should be operable within such limited displays. However, the fact that some users will have access to larger displays implies that these users should not be stuck with using software clearly designed for much smaller displays. This motivates a design requirement that the tools be configurable by the user to suit their available display areas.

HCI takes a wide view and is also concerned with the artefacts that may be used in conjunction with computer-based tools in the achievement of a task, since these can interact and affect task performance. Therefore, it is relevant to consider the on-line provision of support materials for any tools as well as course notes and tutorials. There are a number of formats one could choose including HTML, Adobe PDF or Adobe PostScript. The choice of format should be one that is easily accessible and usable via a web browser — so students can use the documents on-line — yet permits straightforward conversion to off-line format, so the student can mark up the documents or study them when not at a computer.

### 2.4. Environment

The environment in which tasks are performed and tools are to be used can also create design requirements. For ex-

ample in lectures use will often be made of data projectors to facilitate the presentation of computer-based information. These data projectors can vary in their ability to present graphical information clearly. Experience by the authors with one system, for example, revealed that yellow graphics on a blue background was more discernible than white on black. This creates the requirement that the tools be configurable so that colour schemes can be selected to avoid such problems, in conjunction with allowing the user to select a subjectively pleasing colour scheme since individual users can differ in what they view as being an effective choice of colours for rendering information.

Institutions can vary in the facilities they provide and these may differ from department to department. For example in the Department of Electronics and Electrical Engineering at Edinburgh students are provided with access to UNIX workstations. However, computing labs provided by the University-wide Computing Services provide access to PCs and Macintoshes. In addition, student users may have access to computing facilities in the home. This variance of facilities further supports the requirement for multi-platform compatibility raised in S2.3.

## 3. MEETING THE REQUIREMENTS

By observing the design requirements of §2 the authors have developed a suite of Java tools to support the teaching of elementary DSP concepts including Fourier transforms, Fourier series, discrete Fourier transforms, windowing functions, power spectral density, convolution (continuous and discrete), correlation, FIR filters, and rotating phasors<sup>2</sup>. The initial choice of concepts to address was guided by teaching requirements for third and fourth year undergraduate Electronics and Electrical Engineering courses at the University of Edinburgh and some tools re-implement MATLAB demonstrations associated with the third year course text, Mulgrew *et al.* [9].

The characteristics of the Java tools are as follows:

- The tools are written in Java 1.1 which has a more straightforward event-handling model than Java 1.0. Although incompatible with early versions of Internet browsers more recent, and freely available, browsers (Netscape 4.6 and Internet Explorer 4 onwards) support Java 1.1 without the need for additional plug-ins<sup>3</sup>. Later versions of Java were avoided since existing browsers fall behind in their support and so require the Sun Java plug-in.
- The tools can also be run as stand-alone applications under a suitable Java environment.
- To support rapid familiarisation with the tools menu-accessed help information about what each tool does and a summary of the operations supported is available. In addition, the tools feature a context-sensitive help system — moving the mouse over an interface object causes an explanatory message to be displayed.

<sup>2</sup>These tools are accessible via the WebEng web-site.

<sup>3</sup>Macintosh users require the Macintosh Runtime for Java and, possibly, the Mozilla MRJ Plug-in. This is a side effect of the limited support that Macintosh Internet browsers have for Java rather than any problem specific to the DSP tools.

- The user can change the colour schemes of the tools, choosing the colours of the components of graphical objects. For example users can set the colours of the axes, grid-lines, background and data sets of a graph.
- Animation is exploited where appropriate. Some animations are under complete user control allowing the user to pause the animation to examine points of importance before continuing.
- The tools are runnable on any screen size from 640 x 480 and above, feature re-scalable windows, and are not restricted by limited browser display space when accessed via web-pages.
- Users can change important parameters — for example the point sizes of discrete Fourier transforms — and see the effects of these changes. Discrete Fourier transform and convolution / correlation tools allow the user to sketch their own signals via a simple graph-sketching feature.
- User configuration of the tools is supported via the provision of a simple configuration file. These files — tool-dependent — allow educators to specify the initial settings for colour schemes, graph scales and dimensions, other parameters, and, most importantly, the signals or other data that each tool uses. The configuration files use a simple grammar requiring no knowledge of Java syntax.
- The tools are supported by HTML-based documentation which can also be downloaded in PDF format. PDF is more compact than other document formats (for example PostScript) and readers are freely available. In addition, PDF documents can be structured using hypertext supporting more flexible navigation around a document. Finally, the ability to convert PDF to HTML facilitates the delivery of the same information in both on-line and paper-based forms.

These tools are *not* unique in terms of the concepts that they impart or cover and other researchers and educators in the DSP, engineering and mathematics communities have produced similar tools (see, for example, [3] [4] [10] [11] [12] [13]). However, what makes the WebEng tools unique is their flexibility — the fact that they can be used on a wide variety of platforms and, for many categories of user, without additional plug-ins, unlike, say, the applets of Yeng *et al.* [12] which require a browser plug-in; are not constrained by the available display size; do not have to be embedded in a web page; can be run as stand-alone applications; and support user customisation of the display features. Most importantly, however, what sets these tools apart is the novel characteristic of providing straightforward support for customisability by other educators, a quality lacking in the other Java DSP tools surveyed by the authors. This provides the mechanics needed to allow the tools to be configured by educators at any institution to their own ends without the need for the undertaking of Java programming. It is this feature that the authors hope will, above all others, make the tools attractive to educators in other institutions and to encourage their uptake and use.

#### 4. ASSESSING EDUCATIONAL IMPACT

HCI emphasises the activity of user-centred evaluation as vital to the development of computer-based tools that are both usable and useful. User evaluation can range from showing the potential users prototypes and collecting their comments through to comprehensive usability studies in which the users' task performance, and the computer-based tools effect on this, is analysed in detail.

Having designed and developed some educational tools it is now necessary to see whether the benefits realised by their flexibility arise in practice, to assess whether they are a useful learning aid for students and tutoring aid for educators, to ask the question "are the tools both usable and useful?"

##### 4.1. The Story So Far...

As part of an iterative development strategy the tools have been regularly viewed by lecturers who intend to deploy them in their courses as a teaching tool. Their feedback has guided the design process. This regular consultation was vital since lecturers can provide insight into their courses and how technology might best support the delivery of these courses. This is in addition to their experiences of dealing with students and their knowledge of student needs. Of more interest, however, is the anecdotal evidence arising from the deployment of early versions of the tools as both lecturing aids and as tools available to students over the last two years. In relation to using the tools as a teaching aid one lecturer commented:

*I have noticed that I am asked less questions on the topics covered by the applets. When using the applets in a one-to-one tutorial basis problems in understanding are more quickly resolved, and students more confident than previously when only paper based approaches were used. In fact, students are more likely to ask more in-depth questions than those required for surface learning.*

Via an on-line form students could also provide their views on the tools as a study aid:

*I think that this abstract topic was made a lot easier by the "hands-on" software where I can try what I think the behaviour should be and by that get a better understanding.*

While this evidence is anecdotal the signs are positive, students indicating how the tools facilitate and encourage their understanding of DSP concepts and how this trait is observed by lecturers, who also find the tools of use as an explanatory aid assisting in the resolution of the problems of certain students. Now that the tools are in a more advanced form it is time to embark on a more rigorous and wide-ranging collection of information relating to the usability and utility of the tools.

##### 4.2. Evaluation Questions

It is planned to deploy the tools in the forthcoming October 2000 term as aids for both lecturers and students, exploit-

ing this deployment to gather information to address the following questions:

- Does the use of the tools lead to an increase in understanding of DSP concepts by students? When used by the students as a learning and study aid? When used by the lecturers as a teaching and demonstrating aid?
- If such an increase in understanding arises then, why, and if not, why not?
- How easy is it for lecturers to integrate the tools into their own courses?
- How might the tools be more effectively integrated into a course as a whole?
- In what way can the tools be improved: to support students; to support lecturers?

It is intended that addressing these questions will not only provide information to assist the tool developers and drive forward the iterative development process, but will also provide information of interest to educators relating to how the tools might best be deployed to support the teaching of DSP. This information will also be of interest to other educators, highlighting some of the issues that should be considered when contemplating the development and deployment of computer-based educational tools.

#### 4.3. An Interpretive Evaluation

One problem with devising experiments to assess the impact of software tools on quality of education is deciding upon what sort of data to gather and how to gather this data. Ethical considerations exclude the partitioning of a class into two groups with only one group being able to access the tools. Similarly, comparisons of examination results produced by a group of students who had access to the tools with those of previous years who did not could be contaminated by a number of other factors, not least the variance in the abilities of students or difficulty levels of examinations set from year to year.

With this in mind, the authors have opted for an interpretive approach to evaluation, specifically an exploratory study using on-line forms and student-evaluator dialogue, in addition to interviews with educators, to collect the perceptions of these parties as to the usability and utility of the tools. The nature of the information gathered will, therefore, be qualitative, consisting of comments, views, complaints and descriptions of users' experiences and behaviour. While not yielding data that can be analysed to produce statistical claims relating to improvements in student understanding or lecturer effectiveness, such qualitative data nevertheless produces data from students themselves as to how they feel the tools improve, or, indeed, hinder, their understanding. This data can then be analysed in detail and lecturers can assess whether the reasons given are credible based on their understandings of the subject, course, and student behaviour and learning. In addition, this qualitative data can provide information as to *why* any problems arise in the use of the tools and whether these derive from low-level interface design problems or more fundamental problems relating to the way the tool imparts information

about the associated concept. Knowing why problems arise allows the appropriate re-direction of design efforts [14]. The use of studies based around the collection of qualitative data is common in the HCI field (see, for example, [15] [16]).

User opinions can also provide information on the acceptability of a system [17]. Acceptability is a much more important issue than usability since a system or approach to interaction may be usable without necessarily being acceptable — users often making do with poorly designed systems that nevertheless provide some assistance with their task, and, conversely, neglecting usable systems if they provide little effective assistance. Acceptability is important for educational systems since lecturers will avoid unacceptable tools and stick with more tried-and-tested methods, defending the interests of the students whose educational experiences can determine their future career path.

Collecting qualitative data also allows the elicitation of information relating to the context in which the tools are used as well as the impact of other related artefacts, highlighting contradictions between the way a tool imparts or visualises a concept and the approach taken by course texts, or differences in terminology between the tools and lecture notes. Such information can contribute to determining how various teaching aids can best be designed and improved to support and augment each other to achieve an effective educational experience for the students.

Finally, such an evaluation is relatively straightforward to execute and therefore lends itself to uptake by other educators opting to use the tools. This gives the scope to generate a wide-ranging set of experiences from educators in different establishments as to the effectiveness of the tools and to contribute to the drawing of more accurate conclusions about what educational benefits occur, why these occur, and how these might be enhanced.

#### 4.4. Encouraging Tool Use

A problem recounted by Picone *et al.* is that simply providing tools does not encourage their use. This coincides with the authors' experiences in that while earlier versions of the tools were recommended to students only a small proportion of students actually used them (or, at least, actually completed an evaluation form stating that they had used them).

For any evaluation to be effective it is necessary to avert this problem. One way is to provide more advertisement and encouragement to students at the outset of and throughout a course to convince the students of the potential benefits of trying out the tools. Another approach, cited in §2.2, is to make the use of the tools a requirement of performing course-work thereby ensuring that the students are made aware of the tools albeit in a rather unavoidable way. The authors are considering how best to address this issue.

### 5. CONCLUSION

In this paper issues relating to the development of educational software in Java have been discussed. This development has yielded a set of tools for the teaching of ele-

mentary DSP concepts that are arguably unique in terms of their flexibility. This flexibility has been achieved via the application of HCI principles during the design process, focusing not only on the available technology but also on characteristics of the users, their tasks and their environments. The tools support animation where appropriate and so provide educators with a feature not available in traditional teaching materials such as books or slides. The tools are re-configurable so that educators can customise them towards the requirements of their own teaching activities rather than changing aspects of their courses to conform to the tools. The tools provide context-sensitive help to students — who can have conflicting demands on their time — so allowing more time to be spent using the tools to learn about DSP rather than learning how to use the tools. In addition, students can also interact with the tools and provide their own information thereby facilitating more active engagement and the use of the tools as an aid to completing course-work or understanding examples presented in lectures or books. Finally, in consideration of the wide range of platforms to which users could have access, the tools can be run on a number of operating systems, under the main web browsers, and with support for configuring the interface to differing display sizes.

Responses from both students and lecturers to early versions of the tools has proven positive and there will be an increase in uptake of the tools by lecturers at Edinburgh during the 2000 / 2001 session. To exploit this interest, an interpretive evaluation has been designed and will commence in October 2000. The evaluation will provide information concerning both the educational effects that use of the tools has on students' understanding of DSP concepts and how educators find using the tools to support teaching. The authors look forward to reporting the results of this evaluation.

## REFERENCES

- [1] Eaton, J.W. *Octave: A high-level language for numerical computations*, <http://www.che.wisc.edu/octave/>, Department of Chemical Engineering, University of Wisconsin-Madison, U.S.A, 2000.
- [2] Picone, J., Hamaker, J.E., Brown, R., Hansen, J.L. and Cole, R.A. Modern DSP Education: The Story of Three Greek Philosophers, *IEEE Signal Processing Magazine*, pp48–56, September 1999.
- [3] Chen Lee, H., Crutchfield, S., Hocker, C., Kahn, S., Ma, L., Nesky, M., Rosenbaum, K., Rugh, J. W., Venkataramani, R. and Woo, B. *Signals, Systems Control Demonstrations*, <http://www.jhu.edu/~signals/>, The Johns Hopkins University, U.S.A, 2000.
- [4] Belkhouja, A., Rabenstein, R., and McLachlan, A. *Systool*, <http://www.nt.e-technik.uni-erlangen.de/~anis/systool.html>, Telecommunications Institute I, University of Erlangen-Nrnberg, Germany, 2000.
- [5] Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., and Carey, T. *Human-Computer Interaction*, Addison-Wesley, Wokingham, England, 1994.
- [6] Carroll, J.M., Smith-Kerker, P.L., Ford, J.R. and Mazur-Rimetz, S.A., *The Minimal Manual Human-Computer Interaction 1987-1988*, Lawrence Erlbaum Associates Inc: Hillsdale, New Jersey **3** (2) p123-153, 1988
- [7] Watson, G. *Phasor Animation*, <http://www.physics.udel.edu/wwwusers/watson/phys208/phasor-slow.html>, Department of Physics and Astronomy, University of Delaware, U.S.A, 1997.
- [8] Crutchfield, S.G, and Rugh, W.J. Interactive Learning for Signals, Systems and Control, *IEEE Control Systems Magazine*, p88, 1998.
- [9] Mulgrew, B., Grant, P. and Thompson, J. *Digital Signal Processing Concepts and Applications*, Macmillan Press Limited, London, 1999.
- [10] Huber, T. *Fourier Synthesis*, <http://www.gac.edu/~huber/fourier/>, Physics Department Gustavus Adolphus College, Saint Peter, Minnesota, 2000.
- [11] Hwang, F-K. *Fourier Synthesis*, <http://webphysics.davidson.edu/Applets/TaiwanUniv/sound/sound.html>, Department of Physics, National Taiwan Normal University, Taiwan, 2000.
- [12] Yeng, Y., Brown, .R, Hamaker, J., Weilundemo, T., Duncan, R., and Wheeler, E. *Demos*, <http://www.isip.msstate.edu/projects/speech/education/demos/>, Institute for Signal and Information Processing, Mississippi State University, U.S.A, 2000.
- [13] Tong Zhou, G. and Hong-Jing, L. *Applets for Random Signals and Noise*, <http://users.ece.gatech.edu/users/gtz/ee3340/java/>, School of Electrical and Computer Engineering, Georgia Institute of Technology, U.S.A, 2000.
- [14] Whitefield, A., Wilson, F., and Dowell, J. A framework for human factors evaluations. *Behaviour and Information Technology* **10** (1), pp65–80, 1991.
- [15] Buckley, P. and Long, J. Using videotex for shopping—a qualitative analysis, *Behaviour and Information Technology*, **9** (1), pp47–62, January/February 1990.
- [16] Howard, S. Trade-off decision making in user interface design, *Behaviour and Information Technology* **16** (2), pp98–109, March/April 1997.
- [17] Sweeney, M., Maguire, M., and Shackel, B. Evaluating user-computer interaction: a framework, *International Journal of Man-Machine Studies* **38** (4), pp689–711, 1993.