towards Open Source Software adoption and dissemination

tOSSad

Contract N° 015981

F/OSS Usability Report, Part B:

Tomorrow's F/OSS usability requirements and recommendation for future F/OSS usability research directions

D20

Version 1.0

15 March 2007

Copyright

Version control

Version	Date	Author	Organisation
0.1	01.07.06	Sandra Frings, Julia Velkova, Damiano Versulli	IAT, BIS, PDA
0.2	24.08.06	Sandra Frings, Julia Velkova, Damiano Versulli	IAT, BIS, PDA
0.3	19.09.06	Sandra Frings, Julia Velkova, Damiano Versulli	IAT, BIS, PDA
0.4	08.10.06	Sandra Frings, Julia Velkova, Damiano Versulli	IAT, BIS, PDA
0.5	03.11.06	Sandra Frings, Julia Velkova, Damiano Versulli	IAT, BIS, PDA
0.51	16.11.06	Sandra Frings	IAT
0.52	18.11.06	Uros Jovanovic, Filip van Gool	XLAB, IGB
0.6	20.12.06	Sandra Frings	IAT
0.7	20.01.07	Damiano Versulli	PDA
0.8	28.01.07	Görkem Çetin	UEKAE
0.9	29.01.07	Damiano Versulli	PDA
0.95	09.02.07	Görkem Çetin, Sandra Frings, Uros Jovanovic, Julia Velkova, Damiano Versulli	UEKAE, IAT, XLAB, BIS, PDA
0.96	16.02.07	Sandra Frings	IAT
0.97	23.02.07	Al Harris	Knownet
0.98	24.02.07	Sandra Frings	IAT Finalisation
1.0	15.03.07	Kaan Erkan	UEKAE

Change history

Version 0.1:… Initial document - suggestion for structure

Version 0.2:… Modification of structure

Version 0.3:… Modification of procedure

[1] http://www.gnu.org/licenses/gpl.html
[2] http://creativecommons.org

Version 0.4:… Addition of matrices
Version 0.51:. Integration of all three individual parts
Version 0.52:. Comments by Uros Jovanovic and Filip van Gogol
Version 0.6:… Version to be sent to Damiano
Version 0.7:… Major restructuration of content and overall style
Version 0.8:… Re-factoring the document
Version 0.9:… Re-factoring of Chapter 5; Moved content to Appendix B
Version 0.95:. Discussion on final items
Version 0.96:. Integration of discussion items
Version 0.97:. Final Spelling and English check
Version 0.98:. Finalisation according to partners comments
Version 1.0:… Quality check

Release approval

Name	Role	Date
Görkem Cetin	Technical co-coordinator	26.02.07
Kaan Erkan	Coordinator	15.03.07

Abbreviations

tOSSad:	towards Open Source Software adoption and dissemination
F/OSS:	Free/Open source software
UCD:	User centered design
HCI:	Human computer interaction
MMI:	Man machine interaction
HMI:	Human machine interaction
OMI:	Operator machine interaction
GUI:	Graphical user interface
OSDL:	Open Source Development Labs
MLI:	Mobile Linux Initiative

Table of Contents

1. Executive Summary

The term "usability" is often used as a synonym of "ergonomics" and describes to what extent a product, system, or user interface is usable and complies with defined standards and guidelines. Usability engineering has become a distinct discipline and defines a set of methods on how to engineer usability into product and system development cycles. More specifically, the term usability, in its most general form, describes the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction within a specified context of use.

Within this context, we examine the usability aspects of F/OSS solutions for mobile devices. The reason we have restricted this report to this subject is because we think that mobile devices, in combination with F/OSS, is an aspect which has to be pushed and improved in the future. Usability issues within the context of mobiles devices offer wide challenges.

It is common with mobile devices that the hardware designer provides the software platform, nevertheless, increasing numbers of companies support and use some F/OSS solutions. The usage of F/OSS on mobile devices is increasing and is now at 37% of market share.

We lists reasons and possible solutions of common problems which Linux on mobile devices currently face. Some of these problems are distribution fragmentation, lower performance, power management, multimedia, synchronisation, etc. We describe some mobile devices on the current market, their advantages and problems regarding the usability.

It is also important to note, that mobile applications are different from desktop applications and can not just be ported. There is a shift in goals that mobile applications try to achieve. The users of mobile devices have immediate goals, which need to be resolved by using as little resource and time as is possible. Besides the limitations such as per-minute charges, slow data transfer and poor connectivity there are significant hardware limitations, such as screen size, the means of data entry and the battery life.

We provide a list of the main issues related to the usability of mobile computing together with some suggestion of how to approach issues like selection vs. typing, consistency, user control, design for stability, resilience to failures and icons to clarify concepts, etc. We also give a description of basic graphical user interface (GUI) components and offer recommendations for their use and position on the screen, or in the application, in accordance to the basic data flow of the application.

In order to improve the state of F/OSS mobile applications, we need to identify the current problems. We have gathered three groups of data, in which we outline the problems, solutions to the problem and correlation of the problem to the target group. First, we correlate usability and F/OSS with mobile devices, then usability and mobile devices with F/OSS and finally F/OSS and mobile devices with usability. Related findings can be used to provide Q/A (questions and answers) type of guidelines to interested developers.

We conclude the report with requirements and best usability practices that should

be followed in order to improve the quality of mobile applications. These requirements are complemented with an overview of the current state-of-the-art in combination with Web 2.0 and "push technology".

2. Introduction

2.1. About this report

This deliverable is part of tOSSad project work package 3 (Usability of F/OSS). tOSSad is a Coordination Action (CA) project funded by FP6-IST[3]. The project consortium consists of 19 partners from 15 European countries. The project started on February 1, 2005 and has a duration of 25 months.

The tOSSad project aims at improving the outcomes of the F/OSS communities throughout Europe through supporting the coordination and networking of these communities by means of state-of-the-art studies, national program initiatives, usability cases, curriculum development and implementation of collaborative information portal and web based groupware.

The main objective of the tOSSad project is to start integrating and exploiting already formed methodologies, strategies, skills and technologies in F/OSS domain in order to help governmental bodies, educational institutions and SMEs (Small and Medium Enterprises) to share research results, establish synergies, build partnerships and possibly innovate in an enlarged Europe.

As an FP6-IST project, tOSSad is structured in the following six work packages:

- WP1: F/OSS Study
- WP2: F/OSS National Programs
- **WP3: F/OSS Usability Study**
- WP4: F/OSS Curriculum Development
- WP5: Dissemination and Exploitation
- WP6: Management and Coordination Activities

We have decided to take mobile computing as the focus for this report. We tried to define tomorrow's F/OSS usability requirements by investigating mobile devices, these being the most prominent of innovations over the last 10 years. We believe each mobile device has the potential to carry more open source software, helping the F/OSS paradigm spread.

The main focus of this report is divided into three sections (see chapter 5):

- Usability of F/OSS and Aspects related to Mobile Devices
- Usability of Applications for Mobile Devices and F/OSS Aspects
- F/OSS for Mobile Devices and Usability Aspects

[3] The 6th Framework Programme web page can be reached from http://europa.eu.int/comm/research/fp6/index_en.html.
For more information on Information Society Technologies (IST), refer to http://www.cordis.lu/ist.

3. Usability of F/OSS

3.1. State of the Art

The term "usability" is often used as a synonym of "ergonomics" and describes to what extent a product, system, or user interface is usable and complies with defined standards and guidelines. Usability engineering has become a distinct discipline and defines a set of methods on how to engineer usability into product and system development cycles.

A range of overlapping terminologies have been used by usability and interface professionals over recent years: human factors engineering, software ergonomics, usability engineering, user friendliness, UCD (user centred design), HCI (human-computer interaction), MMI (man machine interaction), HMI (human-machine interaction), OMI (operator machine interaction), GUI (graphical user interface) Design, etc. Throughout this paper, the term usability is to be used universally as a subordinate concept.

Along with the relationship of the term usability to software ergonomics, there is also a strong relationship between usability and usage quality. This implies that the usage of a system is affected by quality aspects. Therefore software quality aspects have to be considered within the context of usability; a term like, usage quality requirements, matches with the tOSSad understanding of usability.

The tOSSad project has conducted open source application surveys related to F/OSS usability and, according to the results, the majority of the respondents indicated that GNU/Linux (used on desktops with a popular GUI like KDE or GNOME), OpenOffice.org and Firefox have enough features and that they are user-friendly. In other words, they have a relatively high assessment of usability.

Apart from the usability aspects the survey discovered an important peculiarity in F/OSS products: there is insufficient information about F/OSS products which are offered as alternatives to commercial versions. This was stated by a small percentage of survey users who started using the products less than half a year ago. That is why it is necessary to intensify informative actions concerning promotion and popularisation of GNU/Linux, OpenOffice.org and Firefox in popular and specialist publications on computer topics, both in print and on the Internet.

In another tOSSad report (Usability involvement in F/OSS projects), we stated that “For a cardinal change of the current status, not only is it necessary to take into account usability standard requirements, but it is critical to adopt them as early as possible in the development process. Such an approach, in effect, can be seen like an adaptation and/or a simplification of the directives specified by current proprietary standards provided by external organisations”.

3.2. ISO/IEC standards related to Human Computer Interaction and Usability

Standards related to usability can be categorised as primarily concerned with:

- The use of the product (effectiveness, efficiency and satisfaction in a particular context of use);
- The user interface and interaction;
- The process used to develop the product;
- The capability of an organisation to apply user centred design.

Appendix A (see chapter 8) reports a detailed list of international standards provided by ISO/IEC. The list is classified in several categories and, for each of them, differentiates between “Principles and recommendations” and “Specifications”.

A deeper analysis of the standards that specifically refers to “usability definition”, and are relevant for this report, is provided in the following paragraphs[4]:

- **ISO 9241-11: Guidance on Usability (1998)**

  This standard (which is part of the ISO 9241 series) provides the definition of usability that is used in subsequent related ergonomic standards:

  > *Usability: the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.*

  ISO 9241-11 explains how to identify the information that is necessary when specifying or evaluating usability in terms of measures of user performance and satisfaction. Guidance is given on how to describe the context of use of the product and the measures of usability in an explicit way. It includes an explanation of how the usability of a product can be specified and evaluated as part of system quality , for example one that conforms to ISO 9001.

  It also explains how measures of user performance and satisfaction can be used to measure how any component of a work system affects the quality of the whole work system in use.

- **ISO/IEC 9126: Software product evaluation - Quality characteristics and guidelines for their use (1991)**

  In the software engineering community the term usability has been more narrowly associated with user interface design. ISO/IEC 9126, developed separately as a software engineering standard, defined usability as one relatively independent contribution to software quality associated with the design and evaluation of the user interface and interaction:

[4] see www.iso.org for all relevant standards

> *Usability: a set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users.*

- **ISO/IEC FDIS 9126-1: Software Engineering - Product quality - Part 1: Quality model (2000)**

ISO/IEC 9126 (1991) has been replaced in the year 2000 by a new four part standard that has reconciled the two approaches to usability. ISO/IEC 9126-1 describes the same six categories of software quality that are relevant during product development: functionality, reliability, usability, efficiency, maintainability and portability (see figure 1):

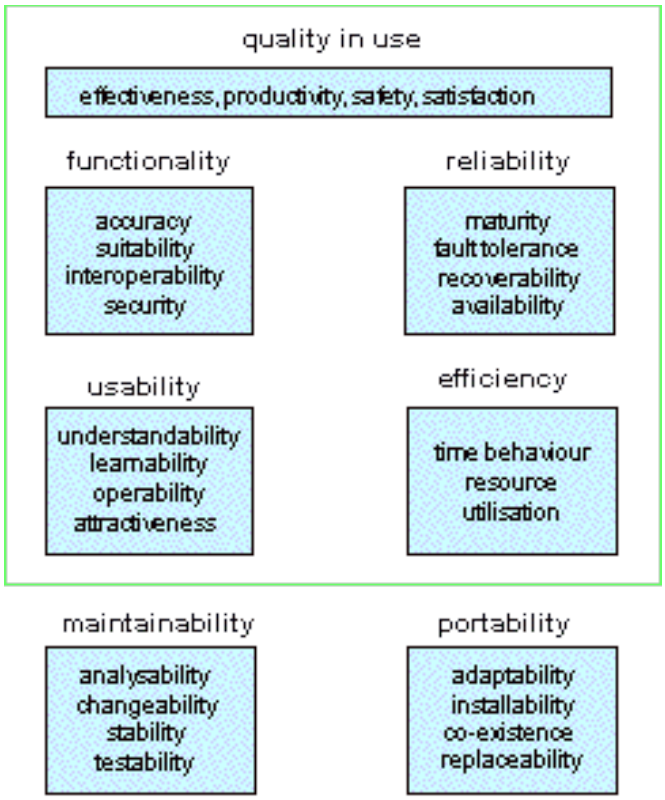

Figure 1: ISO/IEC 9126-1

The definition of usability is similar:

> *Usability: the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.*

The phrase "when used under specified conditions" (equivalent to "context of use" in ISO 9241-11) was added to make it clear that a product has no intrinsic usability, only a capability to be used in a particular context.

The standard now recognises that usability plays two roles, a detailed software design activity (implied by the definition of usability), and an overall goal that the software meets user needs (similar to the ISO 9241-11 concept of usability). ISO/IEC 9126-1 uses the term "quality in use" for this broad objective:

> *Quality in use: the capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use.*

Quality in use is the combined effect of the six categories of software quality when the product is in use. The overall objective is to achieve quality in use, both for the end user and the support user. Functionality, reliability, efficiency and usability determine quality in use for an end user in a particular context. The support user is concerned with the quality in use of maintenance and portability tasks.

Other parts of ISO/IEC 9126 define metrics for usability and quality in use.

- **ISO/IEC DTR 9126-4: Software Engineering - Product quality - Part 4: Quality in use metrics (2001)**

This technical report contains examples of metrics for effectiveness, productivity, safety and satisfaction. Specifying usability requirements and verifying that they have been achieved in a usability test is an important component of user centred design (ISO 13407). ISO/IEC 9126-4 suggests metrics for effectiveness, productivity, satisfaction and safety that can be used for this purpose. The results can be documented using the common industry format for usability test reports, which is included as an example in an Annex to ISO/IEC 9126-4.

- **ISO WD 20282: Usability of everyday products (2001)**

A multi-part standard is being developed to specify the information about usability that should be provided with a consumer product, so that a purchaser can judge the ease of use of the product. It will specify a test method, the characteristics of a "normal user", and how to specify the characteristics of intended users with special needs or with special skills or experience (*Source: Bevan and Schoeffel, 2001*[5]).

- **ISO 9241: Ergonomic requirements for office work with visual display terminals**

ISO 9241 provides requirements and recommendations relating to the attributes of the hardware, software and environment that contribute to usability, and the ergonomic principles underlying them. Parts 10 and 12 to 17 deal specifically with attributes of the software. Parts 14-17 are intended to be used by both designers and evaluators of user interfaces, but the focus is primarily towards the designer.

[5] Bevan N., Schoeffel R. (2001) A proposed standard for consumer product usability. Proceedings of 1st International Conference on Universal Access in Human Computer Interaction (UAHCI), New Orleans, August 2001.

The standards provide an authoritative source of reference, but designers without usability experience have great difficulty applying these types of guidelines. To apply guidelines successfully, designers need to understand the design goals and benefits of each guideline, the conditions under which the guideline should be applied, the precise nature of the proposed solution and any procedure that must be followed to apply the guideline. Parts 12 to 17 contain a daunting 82 pages of guidelines, but even then do not provide all the information for every guideline.

Several check lists have been prepared to help assess conformance of software to the main principles in ISO 9241 (Source: Gediga 1999, Oppermann and Reiterer 1999, Prümper 1999[6]):

- **Part 10: Dialogue principles (1996)**

  This part deals with general ergonomic principles which apply to the design of dialogues between humans and information systems: suitability for the task, suitability for learning, suitability for individualisation, conformity with user expectations, self descriptiveness, controllability and error tolerance.

- **Part 12: Presentation of information (1998)**
  This part contains recommendations for presenting and representing information on visual displays. It includes guidance on ways of representing complex information using alphanumeric and graphical/symbolic codes, screen layout, and design as well as the use of windows.

- **Part 13: User guidance (1998)**
  This part provides recommendations for the design and evaluation of user guidance attributes of software user interfaces including prompts, feedback, status, online help and error management.

- **Part 14: Menu dialogues (1997)**
  This part provides recommendations for the design of menus used in user-computer dialogues. The recommendations cover menu structure, navigation, option selection and execution, and menu presentation (by various techniques including windowing, panels, buttons, fields, etc.).

- **Part 15: Command dialogues (1997)**
  This part provides recommendations for the design of command languages used in user-computer dialogues. The recommendations cover command language structure and syntax, command representations, input and output considerations, and feedback and help.

[6] Gediga, G., Hamborg, K., Düntsch, I. (1999)The IsoMetrics usability inventory: An operationalisation of ISO 9241-10.Behaviour and Information Technology

Oppermann, R., Reiterer, R. (1997)Software Evaluation Using the 9241 Evaluator Usability Evaluation Methods. Behaviour and Information Technologyv.16 n.4/5 p.232-245

Prümper, P. (1999) Test it: ISONORM 9241/10.In: Bullinger H-J and siegler J (eds), Proceedings of HCI International, Munich, 22-27 August 1999.Lawrence Erlbaum, Mahwah, NJ, USA

- **Part 16: Direct manipulation dialogues (1999)**
  This part provides recommendations for the ergonomic design of direct manipulation dialogues, and includes the manipulation of objects and the design of metaphors, objects and attributes. It covers those aspects of Graphical User Interfaces that are directly manipulated, and not covered by other parts of ISO 9241.
- **Part 17: Form filling dialogues (1998)**
  This part provides recommendations for the ergonomic design of form filling dialogues. The recommendations cover form structure and output considerations, input considerations, and form navigation.

- **ISO 9241: Ergonomic requirements for office work with visual display terminals**

  ISO 9241 provides requirements and recommendations relating to the attributes of the hardware, software and environment that contribute to usability, and the ergonomic principles underlying them. Parts 3 to 9 contain hardware design requirements and guidance:

  - **Part 3: Visual display requirements (1992)**
    This part specifies the ergonomics requirements for display screens that ensure they can be read comfortably, safely and efficiently to perform office tasks. Although it deals specifically with displays used in offices, it is appropriate for most applications that require general-purpose displays to be used in an office-like environment.
  - **Part 4: Keyboard requirements (1998)**
    This part specifies the ergonomics design characteristics of an alphanumeric keyboard that may be used comfortably, safely and efficiently to perform office tasks. Layouts are dealt with separately in various parts of ISO/IEC 9995: Information Processing - Keyboard Layouts for Text and Office Systems (1994).
  - **Part 5: Workstation layout and postural requirements (1998)**
    This part specifies the ergonomics requirements for a visual display terminal used in the workplace, requirements that allow the user to adopt a comfortable and efficient posture.
  - **Part 6: Guidance on the work environment (1999)**
    This part provides guidance on the Visual Display Terminal working environment (including lighting, noise, temperature, vibration and electromagnetic fields) providing the user with comfortable, safe and productive working conditions.
  - **Part 7: Requirements for display with reflections (1998)**
    This part specifies methods of measurement of glare and reflections from the surface of display screens, including those with surface treatments. It is aimed at display manufacturers who wish to ensure that anti-reflection treatments do not detract from image quality.
  - **Part 8: Requirements for displayed colours (1997)**
    This part specifies the requirements for multicolour displays that are largely in addition to the monochrome requirements in Part 3.

- **Part 9: Requirements for non-keyboard input devices (2000)**
  This part specifies the ergonomics requirements for non-keyboard input devices that may be used in conjunction with a visual display terminal. It covers such devices as mouse, trackball and other pointing devices. It also includes a performance test. It does not address voice input.

ISO 11064: Ergonomic design of control centres

This eight-part standard contains ergonomic principles, recommendations and guidelines:

- Part 1: Principles for the design of control centres (2000)
- Part 2: Principles of control suite arrangement (2000)
- Part 3: Control room layout (1999)
- Part 4: Workstation layout and dimensions (CD: 2000)
- Part 5: Human-system interfaces (WD: 1999)
- Part 6: Environmental requirements for control rooms (WD: 2000)
- Part 7: Principles for the evaluation of control centres (WD: 2000)
- Part 8: Ergonomic requirements for specific applications (WD: 2000)

3.3. Previous and Current F/OSS Usability Projects and Initiatives

There are a vast number of initiatives, projects and articles about open source and usability. Below we have listed the most significant initiatives currently registered at a worldwide scale. We have tried to find the ones that deal with the most current issues in the field today.

Generally the attention for usability in F/OSS has increased over the past years and thanks to this, much progress has been made. Firefox is a good example, where an open source application successfully competes with proprietary software, though it requires effort from the F/OSS community to face the challenges that still exist.

3.3.1. Better Desktop

Better Desktop is a project dedicated to sharing usability data with Linux developers. Over the past year, many usability tests have been conducted on different parts of the KDE and GNOME desktops. During the project a website was created to serve as a place where developers can watch videos of these tests. By the middle of October 2005, over 200 videos of people using Mozilla Firefox, Evolution, OpenOffice.org, Banshee, F-Spot and other applications can be viewed.

Better Desktop is sponsored by Novell and is part of the OpenSUSE project.

http://www.betterdesktop.org/welcome/?q=data

3.3.2. Extreme Usability

The notion of "extreme usability" has been discussed in the first FLOSS Usability Sprint in February 2005. Usability must be a consideration at all stages of the

software development process and not just during the last steps. Extreme usability is a methodology that incorporates usability in a highly iterative and agile development process and that partners usability practitioners with programmers and users as co-designers and co-developers.

At this event, the methodology on concrete, free (libre) / open source (FLOSS) software projects was reviewed and refined, focusing especially on users from non-profit organisations.

http://www.flossusability.org

3.3.3. GNOME Usability Project

The GNOME Usability Project strives to make the GNOME experience as pleasant and efficient as possible. The project aims both to aid developers in their efforts to create intuitive applications, and to lead by creating designs and detailed mock-ups toward a cohesive and beautiful new generation of the GNOME desktop. The GNOME Usability Project achieves these goals through the creation of an interface guide defining and evolving the GNOME user interface, working with maintainers to find existing interaction problems through user testing, and the visual/interactive engineering of new desktop components. Results in the form of human interface guidelines can be found at the project web site.

http://www.developer.gnome.org/projects/gup/hig

3.3.4. KDE Usability Project

The KDE Usability Project is an initiative to enhance the user experience by assisting developers in creating more usable software. In this project, usability experts work with developers and educate them on integrating usability into the development cycle, reviewing software for usability issues and making suggestions to improve the overall quality of KDE software also in terms of a more user-friendly KDE environment. As the time of this writing, work on the Human Interface Guidelines is continuing with added support from developers through helpful usability surveys.

http://usability.kde.org/

3.3.5. Open Usability

Openusability.org is a project that brings open source developers and usability experts together. There are many usability experts who want to contribute to software projects. And there are many developers who want to make their software more usable, and - as a consequence - more successful. For developers, Open Usability is the first place to get information and to get in touch with usability experts. On the other hand, for usability experts, Open Usability serves as an information resource about current projects and to learn about the responsibilities associated with joining a project.

http://www.openusability.org

3.3.6. OS-Usability: Open Source Usability Information Centre

OS-Usability is an information resource dedicated to providing F/OSS developers

with free and objective information and advice about how to build usable applications. Where possible, empirically validated knowledge is discussed, and where not, no bias to any particular operating system is intended to be shown.

http://forge.novell.com/modules/xfmod/project/?os-usability

3.3.7. Tango project

The project aims to create a consistent user experience for Free and Open Source software with graphical user interfaces. Tango defines a standard icon style guidelines document that applications and desktop environments can adhere to.Inconsistency in the user interface among different F/OSS applications is considered to be a serious gap in the usability, this project aims to resolve this issue and make F/OSS more user friendly. In addition, the Tango icons are licensed under the Creative Commons Share-Alike license. The palette is public domain, it can be used and distributed freely.

http://tango-project.org/

3.3.8. Interesting Articles

In addition to the initiatives reported above, a number of articles have been written about F/OSS usability and related topics.

Here follow the links of the ones that we found interesting**:**

- The Usability of Open Source Software:
  http://www.firstmonday.dk/issues/issue8_1/nichols/#n2
- Usability discussions in Open source development
  http://opensource.mit.edu/papers/twidalenichols.pdf
- Participatory Usability: supporting proactive users
  http://opensource.mit.edu/papers/nicholsmckaytwidale.pdf
- Usability and open source software
  http://opensource.mit.edu/papers/nicholstwidale1.pdf
- Usable GUI design: A quick guide for F/OSS developers
  http://benroe.com/files/gui.html
- Open source Usability: The birth of a movement
  http://www.rashmisinha.com/archives/05_04/open-source.html
- StarOffice Calc v. MS Excel: Improving the usability of an Open Source spreadsheet application
  http://www.sims.berkeley.edu/courses/is271/f01/projects/StarCalc/
- Open source usability is a technical problem we can solve on our own
  http://www.newsforge.com/article.pl?sid=04/07/07/1640244

4. F/OSS for Mobile Devices - State of the Art

4.1. Introduction

The use of Free and Open Source Software (F/OSS) on mobile phones is a rapidly emerging trend, especially within the last few years. This has provoked the development of numerous platforms and applications (aside from Java based) which are all open source but differ in terms of usage on different devices.

The growth in the use of mobile devices with F/OSS installed has revealed a number of problems and issues not present until now. Below is a summary and brief description of the major general issues and problems that currently exist from which we could draw further directions and requirements in the development of F/OSS for mobile devices.

4.2. Brief facts

Some current versions of Linux distributions for mobile devices, used in approximately 37% of all mobile devices[7].

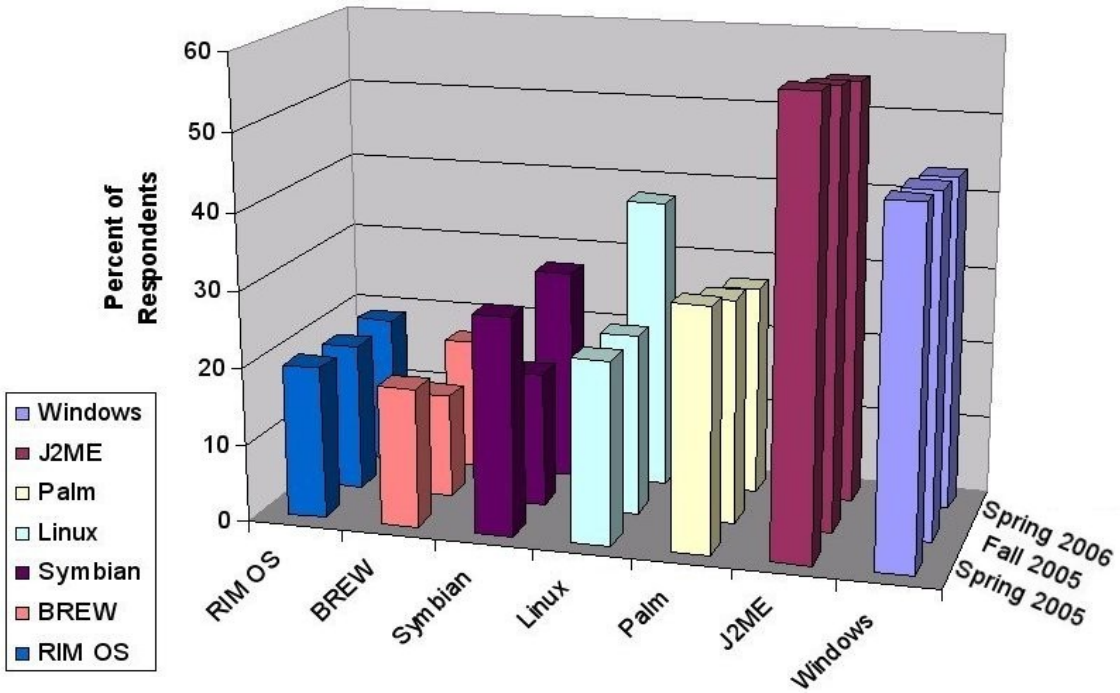

Figure 2: Target Platform for Wireless Applications
(Wireless Development Survey 2006: Spring, 2006 Evans Data Corp.)

[7] Source: LinuxDevices - http://www.Linuxdevices.com/articles/AT7931738328.html

F/OSS for Mobile Devices - State of the Art

There is a constant growth in the share of Linux users. The major manufacturers of devices with embedded F/OSS platform are Motorola, Nokia, Samsung, Trolltech, AccessPalmSource, NEC and Panasonic. All these have various developments: AccessPalmSource has tried to develop a complete SDK, UI and kernel together, since 2004; Motorola runs embedded Linux software (MontaVista) on more than ten models of its mobile phones; Nokia has the 770 Internet Tablet with an own Linux based operating system called Maemo, which is Debian based. An interesting issue with Nokia is that Linux is biting directly on the Symbian market share and that is why Nokia is still reserved about the usage of wider developments of Linux. Samsung has its own platform and is currently testing several phones in the Chinese market; Trolltech (their technology being used by Opera, Motorola, Google Earth) develops the Qt (pronounced “cute”) and provides the GUI for various manufacturers. NEC and Panasonic have worked for some time on the development of another Linux implementation in the FOMA 3G handset for the major Japanese carrier (DoCoMo).

All these efforts for Linux in mobile development and growth of number of users definitely show a constant interest in using F/OSS on mobile devices.

The main reasons for using embedded Linux instead of other operating system appear to be:

- It is open source.
- It has a small footprint (around 2MB for a minimal installation).
- It has no royalty costs.
- It is mature and stable (over ten years of age and used in many devices).
- It is well supported.

Besides mobile phones, another segment in which usability and open source issues appear is in the palmtops. Recently, since the smart phone market is expanding rapidly, the palmtops have become increasingly Internet and network oriented in order to offer similar or even better functionality than a laptop. Currently some of the most popular palmtops are running on free and open source software. These are, for example, Nokia 770, Sony Mylo and Sharp Zaurus.

In terms of platforms used on these devices, they vary for each manufacturer. The most popular platforms are:

- GPE Palmtop Environment
- OPIE
- OpenZaurus
- Qtopia
- Palm OS
- Pocket PC
- Windows Mobile

From these, the first three are based on Linux and are Open Source software. In addition, Nokia is using Maemo and Sharp is using OpenZaurus.

Below is a list with some of the most popular Open Source distributions for palmtops and mobile phones with brief description.

- **GPE Palmtop Environment** - http://gpe.handhelds.org/
  The GPE Palmtop Environment provides an user interface environment for palmtop/hand held computers running the GNU/Linux or any other UNIX-like operating system.
- **OPIE - Open Palmtop Integrated Environment** – http://opie.handhelds.org/
  OPIE is a completely Open Source based graphical user environment and suite of applications for PDAs and other devices running Linux. It is included in various embedded Linux distributions such as OpenZaurus, Familiar and OpenSIMpad.
- **Qtopia** – http://www.trolltech.com/products/qtopia
  Qtopia is Trolltech's application platform for embedded Linux based PDAs, mobile phones, web pads, and other mobile computing devices. Qtopia is shipped on numerous hand held devices including the Sharp Corporation Zaurus line of Linux hand holds, the Sony mylo and the Archos Portable Media Assistant (PMA430), a multimedia device. Qtopia phone edition is expected to start shipping on various smart phones. Qtopia has also been ported to run on the Gamepark Holdings GP2X.
- **Maemo** – http://www.maemo.org
  Maemo is an open development platform to create applications for Nokia 770 Internet Tablet and other Maemo compliant hand held devices in the future. The platform gives developers an easy to use development environment and an end user optimised interface customised for hand helds.
- **OpenZaurus** – http://openzaurus.Linuxtogo.org/
  The OpenZaurus project was started as an alternative Operating System for the Sharp Zaurus Personal Mobile Tool. The original purpose behind the project was to create a system (kernel + root file system) which was a bit closer to what the developer community specifically desired. The method by which this was accomplished was simply to use the Sharp ROM as a base and make alterations, bug fixes, additions and even removals, where necessary, to make the package more open.
- **MobiLinux** - http://www.mvista.com/products/mobiLinux/
  MobiLinux is a Linux-based operating system targeting smart phones. It was announced by MontaVista Software on April 25, 2005. MobiLinux is based on open source and open standard technology, designed for scalability and maximised battery power usage for single-chip mobile phones. Linux 2.6 kernel is used with MobiLinus. It has a rapid boot time of less than one second and an event broker. Its graphical user interface is based on KDrive (also known as TinyX) and GTK+ technology
- **MontaVista** - http://www.mvista.com/products/realtime.html
  This is a version of Linux that has been enhanced to become a fully fledged real-time operating system. The core changes done by MontaVista were made by Robert Love and submitted back to the Linux kernel. Ultimately some of these changes have served as a motivating factor in the development of newer features in the mainstream Linux stable kernel 2.6 series. The work on real-time performance has since continued to a point where MontaVista claims

to support hard real time tasks on Embedded Linux as of MontaVista Linux 4.0

- **LMS** - Linux Mobile System – http://Linuxmobile.sourceforge.net/
  LMS is a full Linux system supported by the new USB Flash Memory Drives. The intention of this system is to boot any PC via the USB port; the user has all the administration and analysis applications selected, without needing to install the application.

4.3. General problems

Below is a list of the most common challenges and problems which Linux on mobile devices currently face. Each problem is stated and briefly explained accordingly.

- **Fragmentation:** This is perhaps one of the major problems confronting all manufacturers of embedded Linux platforms.This problem is the basis of most of the rest of the problems stated below. Currently there are many Linux distributions for mobile devices: MontaVista, Maemo, Zaurus, etc, causing incompatibility problems. This wide diversity of applications, specific to the individual phone or mobile device, mean upgrades or further developments of the software is very difficult to control and implement. Today, silicon providers, Linux distribution houses, ISVs, and OEMs can offer, deploy and maintain divergent versions of the Linux kernel and stack, complicating integration and application development. The most useful slogan here is “Consolidation, not fragmentation”.
- **Lower performance:** Most of the devices with embedded Linux are of significantly lower performance compared to the competing operating systems. This directly reflects the reduced usability of the device with such an OS. Example: Nokia 770 which suffers from latency problems.
- **I/O and Networking:** Many of the devices using embedded Linux have significant problems with data exchange and networking. Such problems significantly decrease the usability.
- **Power Management:** The power management of mobile devices equipped with embedded Linux is not standardised across mobile devices. The OSDL desires to see either a unified cross-processor power and energy management scheme, or a mainstream high-level "umbrella" that covers embedded, desktop and even blade-based thermal management.
- **Security:** Currently, there are many ways to hack the device, even at the baseband level, characteristics deemed undesirable by the carriers. There are initiatives to bring trusted computing to mobile devices, however it will take time to bring these initiatives to market.
- **Multimedia:** The devices have poor multimedia feature sets and those they have are often buggy. Increasing numbers of applications are ported to run on Linux based mobile devices but the fragmentation problem slows down the development process and makes it difficult to respond as rapidly as the manufacturers and developers of proprietary software are able to do.

- **Capability main streaming:** While hardware and software support exists to

build and deploy Linux phones, those capabilities are not main stream, in that the kernel versions, patches, device drivers and other software are not part of the main Linux source tree. Even minor forks raise costs and can limit interoperability.

- **Synchronisation:** There are various problems reported with synchronisation of the mobile devices and computers. The main problem seems to be the MS Outlook (Calendar & Contacts) synchronisation which is claimed by many users as buggy and difficult to manage. With some mobile phones, users also complain about the low usability of the Organiser applications which have very poor usability and exibit generic mistakes (e.g. lack of display of today's date and poor navigation in Sharp Zaurus).

Below are some examples of usability problems with some of the most popular PDAs with F/OSS – Nokia 770, Sony Mylo and Sharp Zaurus.

4.3.1. Nokia 770

It seems that this Nokia palmtop, dedicated primarily to the Internet user, suffers a classical usability problem, latency. Many users have complained that the device reacts too slowly and this significantly decreases its usability. The Nokia 770 is based on Maemo (http://www.maemo.org) open source platform with the addition of third party codes like Opera (fully featured, not like other PDA browsers) and others. Speed is not a problem if battery life is good, and since many applications have been ported to this platform, especially from GPE project, there are already many PIM applications for it. Another problem seems to be synchronisation and calendar / task management options: the built in software is claimed, by many users, to have poor usability combined with missing features and buggy clients. In addition, some users complain about the text input which is “painful”. Webmail works well on it, but sending a message with their text input takes a lot of effort.

Figure 3:Nokia 770 PDA

Solution of the latency problem. Below is a list of bullet points offering some

solutions to this problem.

- Wherever possible, use multi-threading to push latency into the background.
- Latency can often be hidden from users through multi-tasking techniques, letting them continue with their work while transmission and computation take place in the background.
- Reduce the user's experience of latency.
- Acknowledge all button clicks by visual or aural feedback within 50 milli-seconds.
- Display an hourglass for any action that will take from 1/2 to 2 seconds.
- Animate the hourglass so they know the system has not died.
- Display a message indicating the potential length of the wait for any action that will take longer than 2 seconds.
- Communicate the actual length through an animated progress indicator.
- Offer engaging text messages to users, to inform and entertain while they are waiting for long processes, such as server saves, to be completed.
- Make the client system beep and give a large visual indication upon return from lengthy (>10 seconds) processes, so that users know when to return to using the system.
- Trap multiple clicks of the same button or object. Because the Internet is slow, people tend to press the same button repeatedly, causing things to be even slower.

4.3.2. Sony Mylo *(MY Life Online)*

Figure 4:A picture of Sony Mylo

This palmtop came out on the market in Q3 2006 and is too new to be discussed in terms of usability, but the first reported problems are:

- The screen is not big enough.
- The device is not intuitive enough – for this reason, the device was successful in Japan (where the buyers read the manual before using the device) and failed in the U.S. (buyers do not read the user manual and considered the interface not intuitive enough).
- Sony Mylo runs on Linux which, by rumour, is Wind River Embedded Linux

with Qtopia as the application development environment.

4.3.3. Sharp Zaurus

The Sharp Zaurus is the name of a series of Personal Digital Assistants (PDA) made by Sharp Corporation. One of the most notable products of this series was SL-5500, which can run Qtopia[8], OpenZaurus or OPIE. It has a docking cradle and stylus.

Below are listed some of the major usability problems with the Zaurus:

- End-users complain that the SL-5500 is very Linux-like: “it's got a ton of cool features, but they are poorly integrated.”
- There are problems with mouse gestures (click and drag). Examples: Poor integration of calendar features. Similarly, if the user clicks the calendar button on a computer running PalmOS, the calendar immediately opens to show today's appointments: after pressing the button, user cycles through views of the current week, the current month, and so on. Likewise, repeatedly pressing the Calendar button on the SL-5500 causes the Calendar application to cycle through the day, week and month views - but the developers forget the critical step of instructing the application to show the view for the current date. As a result, when the user clicks the Calendar button, he needs to manually check to make sure that the Zaurus is actually displaying today, as opposed to next week. Even worse, there is no "today" button so the user should choose the "today" command from the "View" menu.)
- The device is inadequate in terms of usability locations of certain menu options.

[8] Qtopia is Trolltech's application platform for embedded Linux based PDAs, mobile phones, web pads, and other mobile computing devices.

5. F/OSS, Mobile Devices, Usability: cross evaluation

5.1. Introduction

One of the main goals of this paper is to discuss future F/OSS usability research. It is well known that usability, alone, is a particularly wide field of research and an exhaustive analysis would surely need a lot of effort. F/OSS can also be a very challenging field of research..

In order to focus our findings and, as a side effect, to narrow our scope, we decided to give a critical role to the word “future” and to try to figure out what is going to happen in the years to come.

Even if we strongly believe that current trends in F/OSS usability will grant an increasing adoption of F/OSS based desktop computing, we considered that an even bigger dissemination of F/OSS technology could be achieved by the way of the so called “embedded devices” and, specifically, through PDAs and smart phones. Considering the high number of PDAs and mobile phones that are currently in use, and the miniaturisation processes that are bringing increasing processing power and capabilities to the next generation mobile devices (so-called “smart phones”), it is really not hard to predict that in a few years several hundreds of millions of people could have a small, specialised, computer in their pocket, to be used in the palm of a hand.

With such an assumption, we decided to focus on the three concepts

- Usability
- F/OSS
- Mobile Devices

and attempt to understand how they relate to each other.

In figure 5 we explain the process we chose: after presenting information about international standards referring to usability (section 3.2), we briefly discuss design guidelines to be adopted in the development process of mobile applications (section 5.1) with some highlights about the adoption of different kind of “components” in the mobile context.

Then we focus on the three areas mentioned above, trying to evaluate how each area will impact the scenario determined by the other two. In other words, given the set obtained by the three entities we researched

{'Usability', 'F/OSS', 'Mobile Devices'},

we analyse how each component can affect the subset formed by the remaining two:

- Section 5.2: 'Mobile Devices' ⇨ {'F/OSS', 'Usability'}
- Section 5.3: 'F/OSS' ⇨ {'Usability', 'Mobile Devices'}
- Section 5.4: 'Usability' ⇨ {'F/OSS', 'Mobile Devices'}

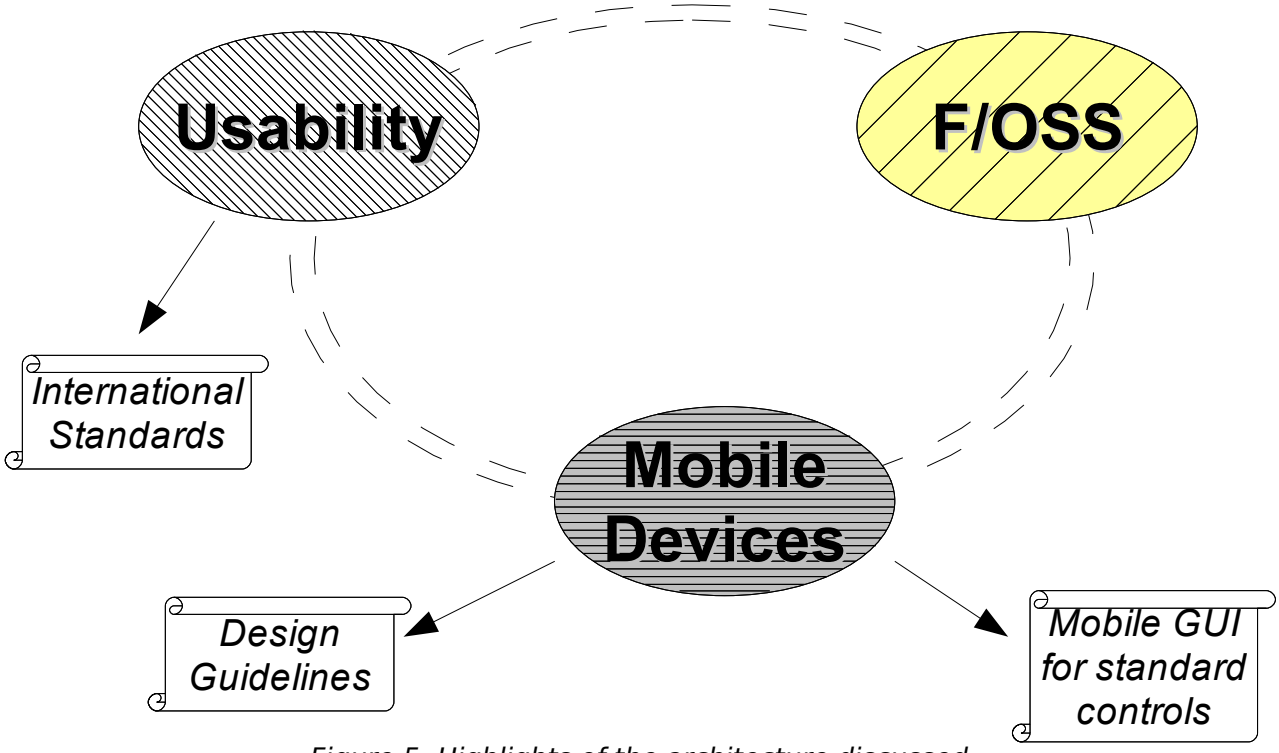

Figure 5: Highlights of the architecture discussed

5.1.1. Design Guidelines for Mobile Applications

Whether in the back seat of a taxi or walking down the street, people are likely to need their hand-held devices to perform in distracting situations. In restaurants, meetings or elevators, hand-held users are likely to be distracted and in a hurry, so designs must include context and forgiveness. In the context of web applications, mobile devices are more about instantaneous search and retrieval while desktops accommodate surfing for hours. Table 1 briefly compares the typical desktop usage to the typical usage of a mobile device. The stakes for mobile usage are higher than for desktop: per-minute charges apply, data transfer is slower, and connectivity is sketchy. Mobile users may be using their leisure time to gather information, but they typically have immediate goals.

Desktop Web	Mobile Web
Comparing prices of flights and making reservations	Checking status of a particular flight
Gathering background information on a company including maps	Getting driving directions to a company while on the road
Researching a medical condition	Monitoring a medical condition
Reading a movie review and/or watching a trailer	Purchasing a theatre ticket to avoid the line

Table 1: Comparison of use of desktop and wireless web (Source: Weiss 2002)

F/OSS, Mobile Devices, Usability: cross evaluation

In the following paragraphs the main issues related to mobile computing usability have been reported together with some suggestion about how to approach such issues:

- **Select vs. Type**: Typing on a hand held device is extremely difficult unless a keyboard is attached. However it is not always feasible to attach a keyboard. Moreover alternative input methods such as handwriting recognition on screen keyboards are slow.
  - Offer selection mechanisms rather than requiring typing. Exception: number of choices is too high for easy selection (e.g. 20 choices and more)
- **Be Consistent**: Users do not want to learn new techniques to access information unless the new techniques will save them a great deal of time and effort.
  - Borrow from well-designed applications when user interface standards and guidelines are either not available or are not yet developed enough to support your interaction mechanism
  - Do not invent user interfaces when one of the existing interfaces will do nicely
  - Use the same terminology and interaction schema within the same application and between applications (this will reduce the learning curve for new features)
- **Consistency between Platforms**: Consistent porting applications from desktop to mobile can hinder good design.
  - Redesign from the mobile user's perspective is required
- **Imply User Control**: Due to the small screen size, users of mobile applications often are guided through pre-organised menu lists while desktop user interfaces allow users to manipulate more freely by clicking and dragging objects.
  - Mobile menus can be designed to imply user control
    - Provide cross linking between areas of applications
    - Give the ability to cut and copy data and paste it elsewhere
  - Enable common actions on visible data, e.g.
    - Provide access to e-mail when a user selects data including an '@' symbol
    - Enable users to call when they have selected a number that correlates to common dialling systems.
- **Design Stability**: When a network connection drops, the application should restore state and context once the network goes back online:
  - Display a message such as "parameters restored"
  - Even offer to restart the process, but if so the data from earlier steps should pre-populate entry fields
- **Provide Feedback**: Each screen of an application should provide the user with enough information to understand what the application is and how to navigate from that page
  - If the display is too small to provide a detailed explanation, a clear path to

that information must be available either from a soft key menu or the devices menu key

- **Forgiveness**: If a user makes a mistake, the user interface must offer a means to correct it.
  - Offer a “back” button
  - Alternatively the menu key could support “undo”
  - Also multiple level undo is preferable to single level undo
- **Use Metaphors**
  - Use metaphors from the real world
  - Instead of referring to a place for storing information as a “repository”, one can call it a “file cabinet” or a “folder”
  - The bookmark is a very effective metaphor for the wireless web, especially since entering URLs into a mobile device is so awkward
- **Clickable Graphics Should Look Clickable**
  - Images that are static and not linked should not appear clickable
  - Clickable layout means that they should have defined borders and/or should have high contrast with the background colour
- **Use Icons To Clarify Concepts:** Icons provide users with additional assistance.
  - Design your icons as representations of concepts, not as hieroglyphics (pictograms that combine thoughts by overlapping images from the real world)
  - The best icons are very simple representations, usually nouns
  - Verbs are more abstract and are difficult to convey as icons (such as “connect” and “disconnect”)

5.1.2. Mobile GUI standard “Controls”

Although mobile devices come in many sizes and shapes, several user interface constructs are common to graphical user interfaces of all mobile devices. This section steps through each user interface control and provides recommendations for use. Concepts presented here are based on result from Weiss, 2002[9]

- **Icons**: Icons are graphical representations. There is no standard for size, colour depth, or style among mobile devices. The following are common sense principals
  - Keep icons small, but recognisable. All hand held devices have small displays and although your icon may be beautiful it takes up valuable screen real estate. However, if your users cannot tell what the icon is meant to represent, it has a detrimentall affect on the user interface.
  - Use high contrast colours. Mobile devices are used in different lighting conditions, in sunlight they can be “washed out”.
  - Make icon edges stand out. Use black if you can. Thick lines are easier to

[9] Weiss, Scott (2002): Hand held Usability. John Wiley & Sons LTD, 2002

identify in soft light.

  - Avoid three dimensional icons. 3D looks great on the desktop where pixels are in abundance, but it loses its allure on the small screen. Keep your icons flat.
  - Keep icons simple and avoid combining pictures to make a “sentence”.
  - Nouns are easier to iconify than verbs. A “paper clip” is easier to conceptualise than “attach”.

- **Audicons**: Audicons are the sound equivalent of icons. Brief distinctive sound effects can add usability and enjoyment to your application. Audicons are an effective way to inform your user of a task’s completion or success in a direct manipulation task. Sound in applications should never be the only user interface, as many users will mute their devices.
- **Menus**: All mobile devices support menus of one type or another. Menus are sets of commands that are hidden behind a graphical button, menu bar, or tap-and-hold paradigm. Menu commands affect the entire application or portal that they cover.
  Menu bars are an interactive, hierarchical command structure that enables a user to navigate within an application. On PDAs, the menu bar is often accessed through the menu hardware key or by tapping in the title of the screen. On phones, the menu is often accessed through a dedicated hardware key.
- **Popup Menus**: Popup menus appear in place over a call to action. The “call to action” typically has a push button like appearance with an arrow or triangle image superimposed. Upon a button or a stylus tap, the menu appears and enables the user to select an item. Popup menus are often used to identify the content adjacent to them, this function being similar to a compressed set of radio buttons.
- **Text Entry Fields**: Text entry fields are boxes or dotted lines where text must be entered by the user. They usually have a text label above them or to their left. Most user interfaces indicate when a text entry field is active by way of enhancing the border or showing a blinking insertion point.
  Different products support different filtering for text entry fields, such as quantity numerical, text or password. Quantity filters beep after certain number of input characters. Numerical and text filters typically beep and otherwise ignore unacceptable input. Password filters have several implementations, but almost always change character inputs to asterisks (“*”) as the insertion point moves forward.
- **Check Boxes**: A check box is an on-off control. Check boxes may appear alone or in groups, but in groups check box functions are unrelated – one check box’s state does not affect the state of any of the others. Check boxes appear adjacent to text or graphical labels. Some user interfaces overload check boxes to show an intermediate state, often as a grey fill rather than a blank or “X”. Use your judgement – such user interfaces are rarely ideal.
- **Radio Buttons**: Radio buttons always appear in groups, since they are exclusive sets of on-off controls. Only one radio button in a group may be on at a time. A radio button can never appear alone – in that case, a check box would be more appropriate.

- **Push Buttons:** Push buttons look and work the same as they do on the desktop. They are metaphorically similar to hardware push buttons. They initiate an action when clicked. When including push buttons in your user interfaces, follow all of the guidelines from labelled and soft key buttons. In addition follow the conventions on feedback, naming, placement and state as discussed below.
  - Provide audio and visual feedback indicating a button press during the press process
  - Name buttons according to the action they perform, rather than “OK” or “Yes”.
  - Place buttons at the bottom of the display, left justified, when possible
  - Place the most logical action button to the left
  - Place the “Cancel” button rightmost
  - Buttons can be active or inactive. Inactive buttons are typically dimmed in some fashion.

5.2. Usability of F/OSS and Aspects related to Mobile Devices

In the following list we discuss typical problems that are encountered within the F/OSS community when dealing with usability issues. We will focus on some of those problems and we try to figure out the consequences of the addition of mobile devices to such a scenario. Furthermore, for each problem, we make some recommendations and/or suggestions, directed toward to the development community, in order to limit the inefficiencies of development activities within F/OSS technologies.

- **Problem n° 1: F/OSS Usability is not a technical problem.** It's easy, for F/OSS developer, to solve “technical” problems. They have proper skills for this. But, from another view, usability is not a technical problem: it requires “social competence”, “interaction”, “interface design”, and lots of other competences that are not easily found within the technical community (system analysis, programming, etc.). When considering mobile devices, a developer tries to solve a specific problem to gain some information of personal interest. The usability is not of primary concern and comes as an afterthought to the developer, because the developer knows all the quirks of the program, and paying attention to the details of some mobile platform is time consuming, even more than on the desktop platforms. Developer's personal preferences usually come before the user's as usually the program is an answer to some personal need

Recommendation/Suggestion:
For the developer interested only in solving the technical part and not being concerned about the spread of usage to more wide use, it would be impractical to force them to follow the rules of improved usability. For the developer wishing to make a program for wide usage, the first rule should be to learn about the specifics of the mobile platform one is working on, and to pursue the rules of the platform throughout the whole process and in every possible aspect of the program. Due to the specifics of each mobile platform, the advise would be not to try to enforce the known practice of developing desktop solutions, but to contact the developers of the mobile platform to gain knowledge about the tool philosophy

- **Problem n° 2: General lack of documentation about F/OSS usability.** The amount of documentation that is available to most of the development community is low. Even when guidelines are provided (like KDE, GNOME, etc.) their adoption is not easy due to underlying technical complexities that developers are not, typically, oriented to address. When considering mobile devices, this problem increases as the high number of available platforms imply a much higher amount of documentation to deal with. Furthermore, the knowledge built during the last decades to address usability issues in standard desktop based F/OSS applications, will not be fully reusable in the “mobile” context, due to the different environments (screen size, input-devices, etc.).

Recommendation/Suggestion*:*
Build a sort of “pool of knowledge” in the usability field by the way of online books, online references, how-tos, articles, glossaries, etc., dealing also with mobile devices.
Contrary to several current ISO/IEC standards, such a “pool of knowledge” should be available on an “open” model, under free-content-licences (GNU FDL, Creative Commons, etc.).
Developers of successful applications or applications that are considered to follow good usability principles should be encouraged to share their knowledge and expertise, by the way of an online “F/OSS usability hub”

- **Problem n° 3: F/OSS usability is not properly supported/funded/backed by private companies and public institutions.** With few exceptions (Novell, SUSE, RedHat and few universities), usability research is not ranked as a high-priority by most private companies and public institutions. Furthermore, software giants (Microsoft, Oracle, Symantec, etc.) are typically approaching usability research in a self-satisfying way: to improve their own products and, at the same time, to limit the amount of knowledge released by the company. When considering mobile devices, an increasing role is played by mobile phone manufacturers and the platform they choose as the operating system for their devices. Even if a number of manufacturers are considering F/OSS technologies

for their products, at the moment their involvement is still limited to "development guidelines" for their products.

> ***Recommendation/Suggestion***:
> *Government and public bodies should promote usability research especially when applied to the development of F/OSS applications, so that the knowledge created can be spread throughout the development community and, as a side effect, this research could be conducted without the typical constraints that marketing and sales departments of private companies typically push on to the development team.*

- **Problem n° 4: F/OSS usability ranks low as a cross-field research:** Usability studies have often been conducted by computer scientists or engineers with the aim to enlarge the user base of a technology (personal computers, mobile phones, PDAs, etc.) and software applications (browsers, word processors, etc.). But usability is not strictly coupled to technical issues but it is connected to issues like information architecture, visual design and other not-strictly technical fields. When considering mobile devices, usability becomes an even broader field, for example, the range of skills and ages of people using mobile devices is much wider compared to the users of personal computers.

> ***Recommendation/Suggestion***:
> *Some efforts should be spent in order to start usability research as a cross-field research between different faculties or universities.*

- **Problem n° 5: F/OSS usability does "cost":** a big part of the development community clearly recognises that considering usability within their project means a general increase in the cost of the project, if not in terms of money, surely in terms of development time and project management. When considering mobile devices, due do the nature of the problem, mobile manufacturers clearly recognise usability as having a strong value for their devices. Some manufacturers even prepare usability guidelines for their products. But even if they recognise the value of usability, they do not have a strict focus on F/OSS, at least in terms of applications and/or user interfaces. Nevertheless, due to the very high number of mobile devices, they can play a very big role in terms of dissemination of both usability and F/OSS concepts.

> *Recommendation/Suggestion:*
> *Developers, especially F/OSS ones, need to know that usability can heavily improve their applications and related adoption. Such improvement needs to be recognised as the counterpart of the overheads brought by usability: increased development time, increased rate of communication level, higher integration complexities.*

Developers of mobile platforms should highlight the design principles of their platform and also pinpoint the best applications developed for their platform, so that the rest of the developers can gain knowledge and recipes on how to approach the problem of usability.
Due to the nature of F/OSS development, developers interested in the usability principles should be allowed to take over the non-technical development and apply their knowledge to improve the usability of the program.

- **Problem n° 6: Wide range of F/OSS applications and user profiles.** The wide adoption of computing-technologies in modern life, bring an increasingly wider user target base and user profile (skills, competencies, ages, behaviours, expectations, etc.) to the development community. Mobile devices are used by a wide range of users and, as such, are going to increase the necessity to properly identify the right set of users, application functionalities, and system application usability.

***Recommendation/Suggestion**:*
Early in the development process, it is needed to classify the application and to identify the right target for it. Only after this, it will be possible to correctly define the usability requirement list and properly start the development activity.

5.3. Usability of Mobile Applications and F/OSS aspects

In the following list we to present nine typical problems that developers need to face when developing applications for mobile devices. Then we try to figure out if, and how, the F/OSS development model could make a difference to such a scenario. Recommendations and/or suggestions are provided in order to limit the risk factors presented.

- **Problem n° 1: “Mobile users” not experienced as “computer users”:** People generally have a lot less experience of using their mobiles online than they do of using their computers online. This means that computer based users can be assumed to have a higher level of existing expertise than mobile and hand held users.

Recommendation/Suggestion:
The complexity of the application logic should be hidden to the user and should be lowered by choosing proper defaults, giving the user no easy way to alter this (otherwise, the complexity would be exposed and the non experienced user is lost again).
Develop easy to use applications so users get used to working with a mobile device.
Imitate the best practices of the applications that are accepted by the public.

- **Problem n° 2: Most web sites are written for desktop computers.** Given the huge amount of applications and interfaces that have been developed for desktop computers, there is a general tendency to adapt such software and interfaces so that they can be used by mobile devices. Unfortunately, they cannot just be transformed into mobile device applications as, doing so, will surely lead to many usability problems.

> ***Recommendation/Suggestion:***
> *Software for desktop computers, which may later be used for mobile devices, need to be developed in the sense of "reusability for mobile devices".*

- **Problem n° 3: Users and usability experts are not integrated in the development process.** As with standard applications, the community developing mobile applications does not involve end users and/or usability experts in the development activity. This is a particularly critical issue as usability cannot be considered as a feature that can be added to a running project. Usability needs to be properly considered from the very beginning of the project.

> *Recommendation/Suggestion:*
> *Apply, to the development process, a user-centred-design approach, include users and usability experts from the very beginning of the project. Usability-hub like "OpenUsability.org" could be an optimal starting-point.*

- **Problem n° 5: Technology comes first.** Many developers prefer to concentrate their efforts on the technological side of their applications, preferring to focus on the capabilities, on increasing the functionalities, on the internal architecture and on the intrinsic beauty of the code, instead of concentrating on the ease-of-use and usability issues.

> ***Recommendation/Suggestion:***
> *Developers should put more emphasis on the development process and should enlarge the development team by including people and/or experts that could bring non-technical skills to the project.*

- **Problem n° 6: Enabling many different features for many different users in many different cultures.** It is common, in software companies, to have sales and marketing departments pushing the technical departments in the effort to enlarge the user base of developed applications. In the F/OSS world, where typically every project is said to be started in order to "scratch an own itch" by the main developer, there is also a tendency to indirectly enlarge the user base by increasing the number of functionalities.

Recommendation/Suggestion:
Every software project, especially F/OSS projects, should carefully evaluate its target group and should really understand the target device (mobile device vs. standard computer) it's going to be used with.

- **Problem n° 7: The physical capabilities of phones are changing quickly.** Mobile devices, and especially mobile phones are evolving at an impressive rate. In the last couple of years the whole mobile scene has registered big structural changes (display resolutions and colours; wireless connectivity; processing power; available memory). If such changes could be documented, in advance, by the manufacturer , the F/OSS developer community would be in a better position to follow platform developments.

Recommendation/Suggestion:
Development activities should be focused on the development of reusable components and on modules that could be easily combined with each one another to form complex applications. With such an approach, platform changes could be easily addressed by re-engineering individual components and not the whole application.

- **Problem n° 8: Limited display size and no keyboard.** Mobile applications need to address many more problems when dealing with user interaction (input/output). Even if display quality is constantly improving, it is clear that the size of a mobile device display will never be comparable even to that of a sub-notebook. Also, although there have been some attempts to put miniature keyboards within the mobile device, it is hard to imagine that such efforts will be appreciated by end users.

Recommendation/Suggestion:
Regardless of the type of applications (proprietary vs. F/OSS), developers need to rethink the approach with end users and should not simply port standard applications to mobile platforms. Applications need to be explicitly developed for mobile devices and the development process needs to include usability tests at every stage of the development activity.

- **Problem n° 9: Varying / unstable connectivity.** When considering online applications, that is, applications that need wireless connectivity to function, mobile devices present several problems in terms of bandwidth and in terms of stability. Even though bandwidth is increasing (from GSM to GPRS to Edge to UMTS to WiFi and to WiMax), mobile users expect to use their devices whilst moving with their cars, while waiting in an elevator or while sitting in a stadium with a thousand other users. There will always be many environmental factors affecting the quality of the signals received and, from the application point of

view, this can be a serious problem.

> ***Recommendation/Suggestion:***
> *When developing application that needs connectivity, try to make offline mode possible. If porting standard desktop applications, re-engineer the network layer to adapt it to the instability of the mobile environment.*

5.4. F/OSS for Mobile Devices and Usability aspects

In the following paragraphs we discuss seven problems that the F/OSS community is currently facing when developing F/OSS applications for the most popular mobile devices. We also investigate how the introduction of usability aspects could impact those problems. When appropriate, recommendations and/or suggestions are given in order to lower the number of potential risk factors.

- **Problem n° 1: Fragmentation and interoperability.** At the time of this writing (January 2007) there are numerous platforms chosen by mobile-manufacturers for their devices. Such platforms are not compatible with one another and also in terms of interoperability many problems are registered. Even when considering F/OSS-based platforms we can count, at least, seven different platforms (GPE Palmtop Environment, OPIE, Qtopia, Maemo, Opensaurus, MobiLinux, MontaVista, Linux Mobile System) with similar compatibility and interoperability issues. When users need to switch from one device to another, such problems bring many difficulties in migrating data and applications from the old device to the new.

> *Recommendation/Suggestion:*
> *The F/OSS community should consolidate platforms to make them more standardised and to lower development process time and costs and to simplify integration and application development.*
> *In order to address the problem of fragmentation, the F/OSS world should develop standard application development platforms that can be used on differing mobile device*

- **Problem n° 2: Performance/Latency.** Some of the F/OSS based mobile platforms seems to be at an early stage of the development activity and present serious issues in terms of performance or latency (e.g.: Nokia 770). These issues heavily impact end user satisfaction; a proper solution should be an high priority.

> ***Recommendation/Suggestion:***
> *From a usability point of view, if latency cannot be avoided, display notifications to the user, hour glasses, beeps, progress indicators, etc. to reduce the bad impression or feeling of latency.*

- **Problem n° 3: Networking and synchronisation.** Current F/OSS mobile platforms have problems in the synchronisation area. Users register problems with organiser and contact synchronisation and program installation.

> ***Recommendation/Suggestion:***
> *This problem could be highly reduced by a consolidation of the platforms (see problem n° 1). Currently, due to fragmentation, developers need to take care of the whole network stack while, with a common-layer, synchronisation becomes the problem of the platform developer and not the application developer.*

- **Problem n° 4: Power management.** Current F/OSS mobile-platforms share problems in terms of power-management. This problem seems to have its root in the desktop F/OSS platform where, for example, the Linux kernel has problems in supporting power management (e.g.: ACPI). on some types of notebooks. Battery life is not unified for all devices running Linux and characteristics like standby and hibernation do not work well.

> ***Recommendation/Suggestion:***
> *Make either unified cross-processor power and energy management schemes, or a mainstream high-level umbrella methods that cover embedded, desktop and even blade-based thermal management.*

- **Problem n° 5: Physical security.** Due to the open nature of F/OSS platforms, there is the very real risk that hardware characteristics are, exposed to developers. This could result in developers easily changing radio frequency characteristics, even beyond regulatory or safety bounds. The end-user could be hacked and exposed to danger without knowing and/or explicitly accepting such a situation.

> ***Recommendation/Suggestion:***
> *Improve physical security. Introduce limitations for modifying the software to avoid change of frequency or other undesirable influences. Protect the internal processes by means of a strict distribution of permissions in the lowest layers system - wide separation of memory, access and input/output rights for processes and applications.*

- **Problem n° 6: Multimedia.** Most of the codecs, technologies and formats currently available on the mobile devices market are patented and can be hardly incorporated into pure F/OSS mobile platforms. Although “Vorbis”[10] is a mature codec that delivers superior performance; few devices and manufacturers support it, even though it requires no licensing fees.

[10] http://en.wikipedia.org/wiki/Vorbis

Recommendation/Suggestion:
Cooperation and collaboration will accelerate adoption of digital audio and video, even as they push the technologies forward. Patent restrictions and proprietary codecs have held back multimedia long enough.

- **Problem n° 7: Capability Main steaming.** Most of the F/OSS mobile platforms rely on a lower layer of components (kernel, patches, device drivers and other software) that are not part of the well known, reliable and supported Linux source tree. Even minor "forks" from the official tree raise costs and can limit interoperability.

Recommendation/Suggestion:
Use a more streamlined development process. Introduce schemes for integration of the changes (patches, etc.) in the main Linux source tree.

5.5. Summary of recommendations

Below is the summarised list of recommendations for the further development of open source software for mobile devices.

- Consolidation, not fragmentation
- Optimised mobile platform
- Streamlined development process
- Improved performance and reduced latency
- Optimise power and energy management
- Implement GPRS and CDMA protocols in low-tier phones
- Implement limitations to avoid hacking
- Unify the software
- Improve interoperability
- Enhance compatibility with older phones

6. Conclusions

The term usability can be explained by several definitions, and in its most basic form, usability defines "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use". Besides usability, one can also use the term quality of use, by which is meant "the capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use". But whatever definition we use, the goal of this document was to identify the elements, that add to the positive experience when using F/OSS solutions on mobile devices.

Although most hardware companies have built their own closed and proprietary software solutions, there is constant growth in the share of Linux users in the field of mobile devices and it is reported that the Linux market share is now at 37%. The major manufacturers of devices with embedded F/OSS platform are Motorola, Nokia, Samsung, Trolltech, AccessPalmSource, NEC and Panasonic. The main reasons for using embedded Linux instead of other operating system appear to be its small footprint, openness, no royalty costs and stability. On the other hand, using F/OSS can also bring negative side effects like fragmentation, as there are many different distributions available; hardware abstraction layers bring lower performance and power management problems; there is minimal multimedia support and there are problems with synchronisation with popular applications..

When evaluating F/OSS on mobile devices, one must first consider some of the general problems of F/OSS that effect the target area. We stated these problems in the chapter about general usability of F/OSS and their relations to mobile devices. We also give recommendations in order to resolve the existing problems. For example, recommendations are given in order to avoid the general problems regarding usability and lack of documentation, etc.

It is also important to realise, that practice taken from desktop application development does not completely apply to mobile applications. The limitations for mobile usage are higher than for desktop: per-minute charges apply, data transfer is slower, and connectivity may be intermittent. Mobile users may be using their leisure time to gather information, but they typically have immediate goals. Besides the difference in application orientation, there are major differences in the hardware: incomplete keyboards, smaller screen and low processing power, etc. With these limitations in mind, we propose several guidelines:

- Selection as opposed to typing, GUI elements with improved user control
- Consistency of application behaviour
- Design for stability, feedback information
- Suggestive graphical information

In order to achieve better usability of F/OSS applications, we emphasise several general improvements in the F/OSS development process, for example, include the users and, if possible, usability experts into the core of the development process. We also propose general guidelines which improve the usability of mobile applications and should be followed by application developers. They are derived

from the fact that mobile applications generally have different goals than desktop applications. Some of these guidelines are:

- Quick access to the most important application functions (few selections only)
- User task flow should be kept fluent and allow images to be used for navigation
- If specific effects are desired, use cascading style sheet (CSS) to control the layout and format elements, such as text and colours. The use of external cascading style sheets is also more informative and minimises code size.
- Avoid frames and void pop-up menus
- As personal information stored on mobile devices is shared, it is crucial to provide easy, consistent and reliable synchronisation mechanism among all applications and devices.
- Enable frequent users to use shortcuts, design dialogues to yield closure, focus of control, etc.

Mobile devices of the future should provide personalised information on the spot, probably with some awareness of the user's current location and context. This information should be provided on demand only and must not interfere with the basic functions of the mobile device (for example, calls should not be interrupted just so that some application can inform you that there is a library 200m down the street).

The document concludes with a list of problems regarding the use of F/OSS tools and F/OSS mobile platforms. We recommend methods to avoid the listed problems. Also, cross-correlation between the F/OSS world and the mobile world is given. These recommendations can be used as quick QA (questions and answers) for the future developers of mobile devices and applications.

We believe, that following the recommended usability guidelines proposed by our document would improve the quality of F/OSS mobile platforms and mobile applications in the future, allowing users to use the new technologies more easily.

7. References

1. The Linux Foundation, formerly known as the “Open Source Development Labs”
   http://www.Linux-foundation.org/en/
2. OSDL – Mobile Linux Initiative (MLI)
   http://old.Linux-foundation.org/lab_activities/mobile_Linux/mli
   (at the moment of this writing, content is going to be moved to the new Linux Foundation web site)
3. Mobile Phones: the Embedded Linux Challenge
   http://interactive.Linuxjournal.com/article/8762
4. LiPS – Linux Phone Standards Forum
   http://lipsforum.org/index.php
5. Nokia S60 UI application requirements templat
   http://sw.nokia.com/id/2a4e3c63-9ed6-486f-9c57-50334a0c3bca/S60_Platform_Requirement_Creation_Guideline_v1_0_en.sip
6. A detailed list of Mobile GUI Environments
   http://www.handhelds.org/geeklog/links.php?category=GUI+Environments
7. A comprehensive list of Maemo applications (used on internet tablets, e.g. Nokia 770)
   http://maemo.org/maemowiki/ApplicationCatalog2006
8. Linux in the Mobile Space: Today and Tomorrow - by Evans Data Corp.
   http://www.Linuxdevices.com/articles/AT7931738328.html
9. GPE Palmtop Environment – http://gpe.handhelds.org/
10. The Open Palmtop Integrated Environment – OPIE
    http://opie.handhelds.org/
11. Qtopia
    http://www.trolltech.com/products/qtopia
12. Maemo
    http://www.maemo.org
13. Opensaurus
    http://opensaurus.Linuxtogo.org/
14. MobiLinux
    http://www.mvista.com/products/mobiLinux/
15. MontaVista
    http://www.mvista.com/products/realtime.html
16. LMS - Linux Mobile System
    http://Linuxmobile.sourceforge.net/

8. Appendix A: ISO/IEC standards and specification related to usability[11]

This chapter gives an overview of which principles and recommendations and which specifications defined within the ISO/IEC are relevant for usability issues.

1. Use in context:

a) Principles and recommendations

- **ISO/IEC 9126-1**: Software Engineering - Product quality - Part 1: Quality model : Software Engineering - Product quality - Part 1: Quality model
- **ISO/IEC TR 9126-4**: Software Engineering - Product quality - Part 4: Quality in use metrics : Software Engineering - Product quality - Part 4: Quality in use metrics
- **ISO 9241-11**: Guidance on Usability: Guidance on Usability

b) Specification

- **ISO 20282**: Usability of everyday products: Usability of everyday products

2. Interfaces and interaction

a) Principles and recommendations

- **ISO/IEC TR 9126-2**: Software Engineering - Product quality - Part 2 External metrics: Software Engineering - Product quality - Part 2 External metrics
- **ISO/IEC TR 9126-3**: Software Engineering - Product quality - Part 3 Internal metrics: Software Engineering - Product quality - Part 3 Internal metrics
- **ISO 9241**: Ergonomic requirements for office work with visual display terminals. Parts 10-17
- **ISO 11064**: Ergonomic design of control centres: Ergonomic design of control centres
- **ISO 14915**: Software ergonomics for multimedia user interfaces: Software ergonomics for multimedia user interfaces
- **IEC TR 61997**: Guidelines for the user interfaces in multimedia equipment for general purpose use

b) Specification

- **ISO/IEC TR 9126-2**: Software Engineering - Product quality - Part 2 External metrics: Software Engineering - Product quality - Part 2 External metrics
- **ISO/IEC TR 9126-3**: Software Engineering - Product quality - Part 3 Internal metrics : Software Engineering - Product quality - Part 3 Internal

[11] Based on informations kindly provided by “Usability Net” at: *http://www.hostserver150.com/usabilit/tools/r_international.htm* (accessed on Feb 2nd 2006)

metrics

- **ISO 9241**: Ergonomic requirements for office work with visual display terminals. Parts 10-17
- **ISO 11064**: Ergonomic design of control centres: Ergonomic design of control centres
- **ISO 14915**: Software ergonomics for multimedia user interfaces: Software ergonomics for multimedia user interfaces
- **IEC TR 61997**: Guidelines for the user interfaces in multimedia equipment for general purpose use

3.Documentation

a)Principles and recommendations

- **ISO/IEC 18019**: Guidelines for the design and preparation of software user documentation

b)Specifications

- **ISO/IEC 15910**: Software user documentation process: Software user documentation process

4.Development process

a)Principles and recommendations

- **ISO 13407**: Human-centred design processes for interactive systems : Human-centred design processes for interactive systems
- **ISO TR 16982**: Usability methods supporting human centred design : Usability methods supporting human centred design

b)Specifications

- **ISO/IEC 14598**: Information Technology - Evaluation of Software Products : Information Technology - Evaluation of Software Products

5.Capability

a)Principles and recommendations

- **ISO TR 18529**: Ergonomics of human-system interaction - Human-centred lifecycle process descriptions

6.Other:

a)Principles and recommendations

- **ISO 9241-1**: Part 1: General Introduction : Part 1: General Introduction
- **ISO 9241-2:** Part 2: Guidance on task requirements : Part 2:Guidance on task requirements
- **ISO 10075-1**: Ergonomic principles related to mental workload - General terms and definitions
- **ISO DTS 16071**: Guidance on accessibility for human-computer interfaces: Guidance on accessibility for human-computer interfaces

9. Appendix B: Usability of mobile applications: Challenges and requirements

This chapter contains several lists of challenges and problems and suggests some recommendations and requirements for the following six topics:

- General challenges and suggestions to improve usability of mobile devices
- W3C Mobile Web Best Practices 1.0: User Interface Guidelines
- Tablet PC User Interface Guidelines
- Usability Guidelines for WAP Applications
- Nokia Mobile User Interface Guidelines
- Guidelines for hand held mobile device interface design

9.1. General challenges and suggestions

- **Problems**
  - most web sites are written for desktop computers
  - Users are not integrated
  - Usability experts are not integrated
  - Technology comes first
- **General requirements**[12]
  - Reaching goal with a few clicks
  - Logical structure
  - Intuitive navigation
  - Should be for non-experts
  - High reliability
  - High speed
  - Possible data synchronisation
  - Minimal weight
  - Maximum portability
  - Appropriate storage options
  - Appropriate size and resolution of the display
  - Components must be optimally tuned to each other
  - Minimal input of characters
  - Application needs to be "usable" even in case of high level of user distraction

[12] http://seap.forum.nokia.com/main/html_readers/usability_is_a_must_in_browser_based_mobile_banking.html

- High level of consistency (e.g. use appropriate guideline)
- Invest in navigation usability
- Design applications for mobile use
- Keep the user task flow fluent and be reasonable with image use
- Provide informative feedback for user actions
- Use efficient and valid mark-up. If specific effects are desired, use cascading style sheet (CSS) to control the layout and format elements, such as text and colours. The use of external cascading style sheets is also more informative and minimises code size.
- Avoid frames
- Avoid pop-up menus
- Perform usability tests (lab as well as real-use context)
- User and usability expert participation throughout the entire development process

- **Mobile future requirements**
  - Push technology
  - 3D and virtual reality
  - Multimedia
  - Bandwidth
  - Wireless Communication
  - Open Source (Nokia example) for greater innovation
  - Focus on human needs and not technology
  - Context awareness
- **Wearable future requirements**
  - Built in multi functional systems
  - Wireless Communication
  - Voice control
  - Price issues

9.2. W3C Mobile Web Best Practices 1.0: User Interface Guidelines[13]

Requirements for mobile Web

- Presentation: layout for desktop size displays not mobile devices
- Help the user to create a mental image of the page - using consistent style
- Input: user input may be very difficult, especially typing an URL
- Bandwidth and cost: mobile networks may lead to long retrieval times which

[13] http://www.w3.org/TR/2006/WD-mobile-bp-20060412/

may be costly

Default Delivery Context (Best Practice Working Group)

- Usable Screen Width: 120 Pixels, minimum
- Markup Language Support: XHTML - Basic Profile [XHTML-Basic]
- Character Encoding: UTF-8 [UTF-8]
- Image Format Support: JPEG, GIF 89a (non-interlaced, non-transparent, non-animated)
- Maximum Total Page Weight: 20 kilobytes
- Colors: Web safe (A Web safe color is one that has Red/Green/Blue components chosen only from the values 0, 51, 102, 153, 204, and 255.)
- Style Sheet Support: External CSS Level 1 [CSS]
- HTTP: HTTP/1.0 [HTTP1.0] or more recent [HTTP1.1]

9.3. Tablet PC User Interface Guidelines[14]

- Keep it simple for non expert users
- Don't rely on users to experiment to discover methods
- Turn the tablet PC, plus your application into an intuitive appliance and include help functions
- Make the link to online help easy to find
- Make the online help context-sensitive
- Consider making online help task-focused, rather than GUI-widget focused, i.e., here's how to do this, vs. here's what that icon means
- Start at the very beginning: The user might simply be frustrated with using the tablet PC. That's not a Windows' problem, or the user's problem. That's your problem to solve, that's if you want a happy customer.
- Modal applications, or applications that change state, aren't intuitive. If the user can see a GUI control, she should be able to use that control. Grayed-out boxes and icons can be frustrating
- Whenever possible, try not to hide the icons or controls that users might want to have
- The drop-down menu is a mouse-friendly construct; for the tablet PC, use intelligent icons for such common functions.
- Make handwriting input fields generous
- With smaller display than desktop monitor and suboptimal lighting there are more problems seeing the screen due to the anti-glare coating on many Tablet PC screens, tablet screens are held more horizontal (facing straight up, or close to it)

[14] "Eight Essentials of Tablet User Interface Success":
- http://www.devx.com/TabletPC/Article/27344/
- http://www.medicaltabletpc.com/content/view/332/29/

- Don't use color to distinguish things, unless color is an essential element
- Because of small displays, hands may be in the way: Try to avoid putting explanatory text next to, or below, a data entry field. Put that text above the field, where the hand is not likely to block it.
- Be sure to test your applications on both left-handed and right-handed users, and be sure to accommodate both sets

9.4. Usability Guidelines for WAP Applications[15]

- Implement navigational menus using a <select> elements
- Keep soft key labels to 5 characters or less
- Use wizards instead of forms
- Keep the content that appears above select and input fields to 1 or 2 lines maximum (including images)
- Assign the most commonly chosen action or most intuitive task to soft keys
- Don't use the <go> task to navigate to a card that is already on the history stack
- Allow the user to dial phone calls from the application by pressing a single key
- Use the format attribute to constrain text input fields to only allow valid character types
- Ensure all decks are smaller than 500 bytes

9.5. Nokia Mobile User Interface Guidelines

Many detailed requirements for Nokia 7710 are provided directly on the Nokia web sites[16].

9.6. Guidelines for hand held mobile device interface design[17]

- Enable frequent users to use shortcuts
- Offer informative feedback
- Design dialogs to yield closure
- Support internal locus of control
- Consistency
  - The “look and feel” should be the same across multiple platforms and devices

[15] Openwave Top 10 Usability Guidelines for WAP Applications: http://developer.openwave.com/dvl/support/documentation/guides_and_references/wap_usability/

[16] http://sw.nokia.com/id/54b57360-4154-4de4-84f4-103888971c0a/Nokia_7710_usability_guidelines_v1_0_en.pdf

[17] http://www.ccs.neu.edu/home/tarase/GuidelinesGongTarase.pdf

  - Elements of mobile interfaces such as names, colour schemes, and dialog appearances should be the same as their desktop counterpart
  - Create input/output methodologies that are device independent - avoid using methods specific to mobile platforms where possible
- Reversal of actions
  - Mobile applications should rely network connectivity as little as possible
- Error prevention and simple error handling
  - Nothing potentially harmful should be triggered by simple operations (e.g., power on/off)
- Reduce short-term memory load
  - Rely on recognition of function choices instead of memorisation of commands
  - Use modalities such as sound to convey information where appropriate
- Design for multiple and dynamic contexts
  - Allow users to configure output to their needs and preferences (e.g., text size, brightness)
  - Allow for single- or no-handed operation
  - Have the application adapt itself automatically to the user's current environment
- Design for small devices
  - Provide word selection instead of requiring text input
- Design for limited and split attention
  - Provide sound and tactile output options
- Design for speed and recovery
  - Allow applications to be stopped, started, and resumed with little or no effort
  - Application should be up and running quickly
- Design for "top-down" interaction
  - Present high levels of information and let users decide whether or not to retrieve details
- Allow for personalisation
  - Provide users the ability to change settings to their needs or liking
- Design for enjoyment
  - Applications should be visually pleasing and fun as well as usable