

Helsinki University of Technology

Department of Electrical and Communications Engineering

Telecommunications Software and Multimedia Laboratory

**Janne Kaasalainen**

**User Interface Design and Usability Testing of  
a Podcast Interface**

Master's Thesis

7<sup>th</sup> November 2007

Supervisor: Professor Tapio Takala

Instructor: Virpi Roto Ph.D.



## Abstract

<b>Author:</b>	Janne Kaasalainen	
<b>Title:</b>	Interface Design And Usability Testing of A Podcast Interface	
<b>Date:</b>	7 <sup>th</sup> November, 2007	<b>Total number of pages:</b> 63
<b>Department:</b>	Department of Electrical and Communications Engineering	
<b>Professorship:</b>	T-111 Interactive Digital Media	
<b>Supervisor:</b>	Professor Tapio Takala	
<b>Instructor:</b>	Virpi Roto, Ph. D.	
<p>This thesis describes the design outcomes and user evaluation of a podcast interface for S60 mobile devices from a user-centric point of view.</p> <p>The design began with a user interface proposal (consisting of a new user interface (UI) style and color schemes) that was shown to public to gather first reactions from a restricted audience. Based on feedback the style evolved and eventually interaction diagrams were made to make sure that the functionality was both possible and logical to implement. The interaction diagram consisted of pre-made visualization mockups of the different screens (made to fit the actual display size of the target device), which included most of the elements to offer the full functionality. The diagrams were then presented to usability experts to be initially evaluated and commented upon.</p> <p>The design decisions were then implemented with Symbian C++ and a prototype was produced to test the theories in practice and to gain information about the users. The implementation itself took advantage of emerging technologies such as 3D acceleration to make it possible to use visually rich user-interface elements and animations. These were produced to help users to understand what is happening within the application as well as offer information that would be hard to show with the traditional user-interface style already established on S60 devices</p> <p>The prototype was then evaluated against already existing software in a user trial. A total of eight recruited to perform various tasks on both of the applications and rate their experiences.</p> <p>The outcomes of the user-study were positive, with the majority of the users (five out of eight) preferring the prototyped interface to the existing and fully working solution. This was considered as a good result given the qualitative feedback from the users, the differences between the technical maturity between the applications (a prototype versus a production quality application already distributed on the Internet) and finally the change resistance of already experienced S60 test participants. Also, some of the findings emphasized the importance of the holistic experience and look-and-feel over a pure set of technical features and merits.</p>		
<b>Keywords:</b> user-centred design, usability, user experience, design, interfaces		

# Tiivistelmä

<b>Tekijä:</b>	Janne Kaasalainen	
<b>Työn nimi:</b>	Interface Design And Usability Testing of A Podcast Interface	
<b>Päivämäärä:</b>	7. marraskuuta, 2007	<b>Kokonaissivumäärä:</b> 63
<b>Osasto:</b>	Tietoliikennetekniikan osasto	
<b>Professori:</b>	T-111 Interaktiivinen digitaalinen media	
<b>Työn valvoja:</b>	Professor Tapio Takala	
<b>Työn ohjaaja:</b>	Virpi Roto, Fl. T.	
<p>Tämä työ kertoo S60 pohjaisille mobiililaitteille toteutetun podcast käyttöliittymän suunnittelusta ja käyttäjätestauksesta.</p> <p>Suunnittelu alkoi käyttöliittymäehdotuksesta jolla kerättiin alustavia mielipiteitä rajatulta kohdeyleisöltä. Saadun palautteen pohjalta käyttöliittymän tyyliä kehitettiin edelleen. Toiminnallisuuden varmistamiseksi muodostettiin interaktiokaavio joka kuvasi sovelluksen rakenteen ja vuorovaikutuksen yksityiskohtaisesti. Kaavio koostui visualisoinneista jotka kuvasivat sovelluksen eri tiloja. Lopuksi kaavio annettiin käytettävyyssiantuntijoille arvioitavaksi ja kommentoitavaksi.</p> <p>Sunnitelmien pohjalta implementoitiin prototyyppi käyttäen Symbian C++ ohjelmointikieltä. Prototyypin tarkoituksena oli toimia pohjana käytettävyydestejä varten ja kokeilla esiteltyjä teorioita käytännössä. Implementaatio hyödynsi mobiilialustoilla uusia tekniikoita kuten laitteen tarjoamaa 3D kiihdytystä rikastamaan käyttöliittymää, jonka toteuttaminen S60 alustan tarjoaman käyttöliittymätyylin pohjalta olisi ollut mahdotonta.</p> <p>Prototyyppiä verrattiin käyttäjätesteissä jo olemassa olevaa toteutusta vastaan. Yhteensä kahdeksan käyttäjää palkattiin testiä varten ulkoisen välitysfirman toimesta. Heitä pyydettiin lopuksi arvioimaan kokemuksensa kunkin sovelluksen osalta.</p> <p>Käytettävyydestien tulokset olivat positiivisia ja yli puolet käyttäjistä (viisi kahdeksasta) koki tuotetun prototyypin miellyttävämpänä verkkisovellukseen verrattuna. Tämän koettiin olevan hyvä tulos huomioiden kvalitatiivinen käyttäjäpalautte ja ero prototyyppien teknisessä valmiustasossa. Osa palautteesta myös korosti käytön kokonaisvaltaisuutta sovellusten teknisten ominaisuuksien yli.</p>		
<b>Avainsanat:</b>	käyttäjakeskeinen suunnittelu, käytettävyys, käyttökokemus, suunnittelu, käyttöliittymät	

## **Acknowledgements**

This thesis is the outcome of my already long-winded studies. The life has taken some unexpected turns till it took me get to this point. My greatest appreciations go to my parents who have had the patience to raise me as with freedom and give the possibilities to follow my interests wherever they have pointed to.

I would like to thank Virpi Roto, my instructor for this thesis, who has provided valuable insight to usability and always provided valuable comments about my work. Also, my thanks go to Tapio Takala for encouraging me and for his guidance.

Lastly I would like to thank people at Nokia Research Center and especially Guido Grassel without whom this thesis would have never started. Another set of thanks goes to Tuomas Tammi who pushed me to actually finish this.

One more to go.

November 7<sup>th</sup>, 2007 in Otaniemi, Finland

Janne Kaasalainen

# Table of contents

ABSTRACT .....	I
TIIVISTELMÄ.....	II
ACKNOWLEDGEMENTS.....	III
TABLE OF CONTENTS .....	IV
<b>1 INTRODUCTION .....</b>	<b>1</b>
1.1 OBJECTIVES .....	4
1.2 SCOPE OF THE THESIS .....	4
1.3 THESIS OVERVIEW .....	5
<b>2 BACKGROUND .....</b>	<b>6</b>
2.1 TECHNICAL BACKGROUND .....	6
2.1.1 RSS.....	7
2.1.2 Podcasts.....	10
2.2 USER EXPERIENCE .....	12
2.2.1 Words about usability.....	12
2.2.2 On user experience .....	13
2.2.3 Notes about emotional communication .....	15
2.2.4 Theory of broken perfection.....	18
2.2.5 Emotional value chain.....	20
2.3 STATE OF THE ART .....	21
2.3.1 iPod, iTunes and iTunes Music Store.....	21
2.3.2 Nokia Podcasting Application .....	22
2.3.3 Other implementations and related work.....	23
<b>3 PODCAST APPLICATION.....</b>	<b>26</b>
3.1 GENERAL DESIGN ISSUES .....	26
3.2 DESIGN DRIVERS.....	28
3.2.1 Role of the application .....	29
3.2.2 Targeted user group .....	30
3.2.3 Appearing simple.....	31
3.2.4 Assisting features.....	32
3.2.5 Assumed use-cases.....	33
3.2.6 Product life-cycle.....	34
<b>4 IMPLEMENTATION .....</b>	<b>36</b>
4.1 INITIAL PROTOTYPES .....	36
4.1.1 Initial feed-reader enhanced to podcasts .....	36
4.1.2 Paper prototypes & interaction diagrams .....	36
4.1.3 Expert evaluations and changes to the final design .....	38
4.2 DRAFT IMPLEMENTATION .....	38
4.2.1 Color scheme development .....	38
4.2.2 The “Wow!” effect.....	40
4.2.3 Emotional functions.....	41
4.2.4 Applying the theory about emotional value chain .....	43
4.2.5 Usability drivers .....	45
4.2.6 Basic usability of the prototype .....	45
4.2.7 Subscribing to podcasts and podcast removals .....	48
<b>5 USER TESTS.....</b>	<b>51</b>
5.1 USER SELECTION.....	51
5.2 TEST PROCESS.....	52

5.3	TEST RESULTS .....	55
5.4	FINDINGS AND THE ANALYSIS OF THE RESULTS .....	57
5.5	IMPROVEMENT IDEAS .....	60
<b>6</b>	<b>CONCLUSIONS .....</b>	<b>62</b>
	<b>REFERENCES .....</b>	<b>64</b>



## 1 Introduction

“Podcasting”: a portmanteau word that combined two words: “iPod” and “broadcasting.” Podcasting is the distribution of audio or video files, such as radio programs or music videos, over the Internet using either RSS or Atom syndication for listening on mobile devices and personal computers. A podcast is a web feed of audio or video files placed on the Internet to download or subscribe. (Wikipedia 2007).

Podcasting has been a rising phenomena from approximately 2004 and is affecting media consumption in ways not many have come to think about before. This is mostly due to the ease of the implementation (the XML formatted files for the syndication feeds are not very hard to write) and the ease of the use. The delivery method is practically invisible to the end user who can in the best case simply click a link on a web page or a specific directory, which directs the XML file to his preferred podcast client or handler.

Naturally, the usability depends on both the client software and the method via which the subscribing is done. There are good implementations and there are difficult ones as well. In some cases this involves creating shortcuts to other software and copy pasting the shortcut pointer to the actual client software manually. This might happen on instances where the feeds are described in HTML header and the browser and the client can not communicate with each other when user selects the feed icon.

Apple iTunes Music Store (iTMS) serves as a good example of a service that goes to some lengths to make the subscription and consumption easy for as long as the user is using their iTunes software. The integration is done in a similar manner as AAC music-selling site, with the exception that the podcasts are mostly free content. The feeds are found via the same methods as the rest of the content and subscribing happens via a button next to the feed descriptions. After this, the

podcasts are updated periodically and new content downloaded in the background while the iTunes media player is running. If needed, the transfers to iPods can be made automatic as well.

This brings up the promise of easiness in the feed usage; the manual work to visit the sites oneself to download the new content is taken away and the user is provided with notifications when the new items are already available to be consumed. He needs not to initiate the downloading which saves his time and effort.

The process can be compared to “Pay-TV” channels where the user buys a subscription to a set of channels. The channels are then made available to him via means such as a set-top box with decrypting capabilities. The user now knows that he has a set of items to be viewed as he wishes. Podcasts differ in the aspect that they are time-shifted and need not to be consumed at specific times. In principle, “time-shifting” means that the content consumption does not need to happen on a specific moment. Also, they are typically free of charge. Nonetheless the basic subscription model applies to both and each relies on the user in a sense that he needs to make a choice what he wants to consume where as public television would offer (or push) all that was available to the consumer.

In fact, the time-shifting properties of podcasting should not be downplayed and that is what makes the user experience fundamentally different from radio (here making no difference if the radio show is transmitted by radio waves or by digital means). Whereas radio can be considered to happen “live”, podcasts are recordings that are passed onwards. These recordings do not need to be played immediately after they have been made public but instead when the consumer is in suitable situation. A recording that has been downloaded while he was asleep can be listened in a car on his way to work on the following morning. In fact, many of the shows produced do not deal with matters that require real-time broadcasting. A talk show sounds the same the following morning as it does at 3 AM when it is aired. Naturally, this does not apply to all content, but nonetheless the content to which it applies is not rare. Instances such as sports are likely to encourage more involving real-time experience than weekly news.

A natural extension to time shifting is location shifting. When the exact broadcasting time of the episode loses its meaning, it is only logical to ask why the content could not be consumed where one wishes. The previous chapter already gave an example where the consumption happens in a car instead of front of the computer or a stereo system. With audio, this is relatively easy to arrange, as listening to audio does not generally require as much of cognitive load from the user as does their processing of visual media. Portable MP3 players can be carried most anywhere and are socially acceptable device in many settings.

The last notable aspect to be mentioned here is the long-tail qualities of podcasting. Most of the new computers of today are more than adequate on audio processing and the recording tools and software are far from expensive. This allows individuals to say, record, process and publish audio with ease. As a result, content that appeals to only a small niche audience of listeners can be made available effortlessly and with minimal costs by active enthusiasts of a given topic. It is theorized that these small audiences combined can form, in fact, a big percentage of the total audience of the given medium (Anderson 2006).

There are negative side effects as well; producing an audio show that has regular updates and interesting content takes much more effort than producing it in textual form. Even further, the text is not necessarily an easy medium either and each of the forms needs some talent from the author to make his content easy and pleasurable to consume. It is likely that not all shows are capable to produce this level of lasting appeal to their audience.

The aspects of time-shifting and location-shifting make podcasting very attractive to be used on mobile devices and more generally on any audio consumer electronics device that people often carry with them. The mobile phone is achieving this status in many countries and provides, in theory, an ideal platform for this sort of media consumption. It has wide options of wireless connectivity and it typically follows with us where we go. This allows it to be used to kill time. The idle moments could well be used to listen the content that has been downloaded to the device earlier.

This gave the incentive to launch a project to see how to provide the best user experience for a mobile podcast client. The initial questions were whether the application should be a part of another application (such as media player or a web browser) or an independent piece of software. The initial attempt was combined to a mobile web browser and given a brief user test about how the participants felt about it. This also served as a technological stepping point to what became the subject of this thesis, to create an individual application that was not restricted by the practicalities of the current environment.

The design for this thesis was started in the beginning of 2006. The end prototype was to concentrate to user experience and usability over technical obstacles and to see if these elements could be enhanced via emerging technologies (in mobile context) such as hardware 3D acceleration. The exact implementation of the clients technical aspects are left out of the scope of this thesis but the possibilities in regards to usability and user experience issues are dealt with in later chapters.

## ***1.1 Objectives***

The focus of this thesis was to create an user interface for a podcast client software that was as easy to use as possible, evaluate the results and demonstrate the benefits that could be achieved by taking advantage of new technical possibilities such as 3D hardware acceleration.

The user interface was to be verified via user trials that tested the produced prototype with non-technical users. The obtained user-test results were then analyzed and suggestions for further development were made. A special effort was made to try to evaluate the pleasantness of the user interface.

## ***1.2 Scope of the thesis***

The thesis concentrates on the usability and user-experience factors on user interface design and proposes a new user interface for a podcast client. Furthermore, the thesis describes the user acceptance tests performed with the prototype that was created to test the interface design.

The technical implementation was done using Symbian C++ and accompanying APIs (such as OpenGL ES 1.1) but the presentation of that is considered to be out of the scope of this thesis. The principles of the design do not depend on software components as such, assuming corresponding ones are available on target platforms.

However, the interaction design of the prototype focused to a particular target platform: in this case a Nokia N95 mobile device. Thus the methods are studied and evaluated in the context of the particular target device and the input/output device features that are available for it. Different devices may have other requirements and possibilities, which make this design heavily device dependant.

### ***1.3 Thesis overview***

This thesis is divided into six chapters. The first chapter introduces the subject and gives a brief background to the project. It also sets the objectives and defines the scope of the thesis.

Chapter two describes the technical background behind RSS and podcasting. It then establishes basic information about user experience. Finally, it summarizes the state of the art in the field of podcasting and introduces two relative applications. Related work is also discussed.

The third chapter introduces the design decisions behind the produced podcast application prototype and establishes an understanding about why some design choices were made.

Chapter four describes the prototype implementation in more detail. It presents how the design drivers and the theories about the user experience affected the outcome. Functionality that was not implemented, or implemented but not to be tested at this stage is also described in this chapter.

The fifth chapter concentrates on usability evaluations and the practicalities of the user-tests. The gained results are described and analyzed.

Chapter six presents the conclusions based on the findings of this thesis work.

## 2 Background

### *2.1 Technical background*

Being hardly more than data enclosures in XML files, the technical concept of podcasting and RSS feeds is very simple and hardly anything that could be called innovative purely in a technological sense. Podcasts, as well as RSS (often called “Really Simple Syndication” but various alternative meanings also exist: “Rich Site Summary”, “RDF Site Summary”), rely on a client that polls the XML file from a server periodically. This file consists of elements that describe the content available either on the file, the site or in an enclosure that is linked to the feed element. The XML file which defines the set of syndicated content is generally referred to as a “feed”. A podcast is a feed that specifically has audio enclosures.

The client application can be either a separate application run independently or included into already existing software, such as a web browser. The client is configured with URLs that point to the feeds. At specified intervals, the client downloads the XML file and checks if it has changed. Such clients usually keep track of what the user has already consumed (e.g. already read or listened to, but as the enclosures can be almost any digital information neither term may be accurate) and visualize that in suitable manner. For textual content this can be done via bolding the new, unread topics or by suitably coloring them. These non-consumed elements are typically referred to as episodes or items, depending on the context. Other terms are likely to exist as well.

After Apple introduced their podcast support in iTunes the summer 2005, they have extended the RSS standard (the RSS 2.\* branch specifically) with concepts such as chapters and album art. A chapter is a marker that defines a specific point of time in the audio track. These chapter marks, combined, form an index that can be used to skip to the interesting position in the audio without resorting to fast-forwarding or rewinding via manual means. Album art, on the other hand, is an

element that describes an image file that is attached to the whole podcast and optionally to each individual chapter inside it. There are no restrictions of what kind of image can be shown (in terms of the content of the image) but they are typically used to describe the show or explain the aspects the chapter tells about. In a sense, they could well be compared to cover pages of textbooks or CD-covers.

Some sites in the Internet have started to collect and catalog the addresses of various podcasts. They often include some ways to filter and search the different feeds via user chosen criteria, such as topics that the podcasts handle (e.g. music or politics). A more advanced method for organizing podcasts has come via OPML (Outline Processor Markup Language) files that can be used to create software that displays to the user the categorizations and offers an easy way to subscribe to the content. All of these methods are generalized as directories, no matter if they need to be accessed via a website or via the podcast software itself.

### **2.1.1 RSS**

RSS is not a specific technology but a category of specifications that have evolved over time. This chapter describes the RSS specification version 2.0.1 (RSS spec, 2007).

RSS is an acronym that stands for “Really Simple Syndication” (other variations of its meaning do exist as well) and it is a dialect of XML (eXtensible Markup Language) RSS files must conform to XML 1.0 specification.

RSS files start with a top level element `<rss>` and end with the closing element `</rss>`. The opening tag has an attribute *version* that describes the version number the RSS file uses.

The `<rss>` element is followed by a `<channel>` element that describes the meta-data about the RSS feed and its contents. The channel tag has a set of required elements it must include:

- Title – This element describes the title of the channel (or feed).

- Link – The URL (Uniform Resource Locator) that refers to the HTML page that corresponds to the channel.
- Description – Phrase or sentence that describes the channel.

The channels can also have optional elements, which deal with various additional aspects. Most notable ones are the <image> and the <item> elements:

- Image – The element has three required sub-elements (url, title and link) and three optional elements (width, height and description).

Item elements have various sub-elements and they are described here briefly as they are the basis for podcast episodes.

- Title – The title of the item.
- Link – The URL of the item.
- Description – The item synopsis or possibly the full text.
- Author – Email address of the author of the corresponding item.
- Category – Includes the item in one or more categories.
- Comments – URL of the page for comments related to the item.
- Enclosure – Describes the media object that is attached to the item.
- Guid – A string that uniquely identifies the item.
- pubDate – Indicates when the item was published.
- Source – The RSS channel the item came from.

The <enclosure> element is discussed here further as it is the basis for transferring the binary content that the podcast applications (and various other clients) use. The element has three required attributes. The first is an HTTP URL that describes where the linked enclosure is to be found. The length attribute tells how

large the enclosure is in number of bytes. The type attribute lists the enclosures MIME-type.

An example of an RSS 2.0 file is presented in figure 1 (wikipedia, 2007).

Figure 1. An example RSS 2.0 file.

```
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <title>Liftoff News</title>
    <link>http://liftoff.msfc.nasa.gov/</link>
    <description>Liftoff to Space Exploration.</description>
    <language>en-us</language>
    <pubDate>Tue, 10 Jun 2003 04:00:00 GMT</pubDate>
    <lastBuildDate>Tue, 10 Jun 2003 09:41:01 GMT</lastBuildDate>
    <docs>http://blogs.law.harvard.edu/tech/rss</docs>
    <generator>Weblog Editor 2.0</generator>
    <managingEditor>editor@example.com</managingEditor>
    <webMaster>webmaster@example.com</webMaster>

    <item>
      <title>Star City</title>
      <link>http://liftoff.msfc.nasa.gov/news/2003/news-
starcity.asp</link>
      <description>How do Americans get ready to work with
Russians aboard the
      International Space Station? They take a crash course in
culture, language
      and protocol at Russia's Star City.</description>
      <pubDate>Tue, 03 Jun 2003 09:39:21 GMT</pubDate>

<guid>http://liftoff.msfc.nasa.gov/2003/06/03.html#item573</guid>
    </item>

    <item>
      <title>Space Exploration</title>
      <link>http://liftoff.msfc.nasa.gov/</link>
      <description>Sky watchers in Europe, Asia, and parts of
Alaska and Canada
      will experience a partial eclipse of the Sun on Saturday,
May 31st.</description>
      <pubDate>Fri, 30 May 2003 11:06:42 GMT</pubDate>

<guid>http://liftoff.msfc.nasa.gov/2003/05/30.html#item572</guid>
    </item>

    <item>
      <title>The Engine That Does More</title>
      <link>http://liftoff.msfc.nasa.gov/news/2003/news-
VASIMR.asp</link>
      <description>Before man travels to Mars, NASA hopes to
design new engines
```

```
        that will let us fly through the Solar System more
quickly. The proposed
        VASIMR engine would do that.</description>
        <pubDate>Tue, 27 May 2003 08:37:32 GMT</pubDate>
<guid>http://liftoff.msfc.nasa.gov/2003/05/27.html#item571</guid>
</item>

<item>
  <title>Astronauts' Dirty Laundry</title>
  <link>http://liftoff.msfc.nasa.gov/news/2003/news-
laundry.asp</link>
  <description>Compared to earlier spacecraft, the
International Space
  Station has many luxuries, but laundry facilities are not
one of them.
  Instead, astronauts have other options.</description>
  <pubDate>Tue, 20 May 2003 08:56:02 GMT</pubDate>
<guid>http://liftoff.msfc.nasa.gov/2003/05/20.html#item570</guid>
</item>
</channel>
</rss>
```

### 2.1.2 Podcasts

The RSS 2.0 specification is in fact sufficient for delivering audio content as enclosures on an RSS channel. The content is not restricted to audio, but for this paper other forms of usage are not discussed.

Podcasts work much like RSS feeds, and there is no practical difference in the basic functionality. The actual client software often presents the XML elements differently and in a manner. The enclosures are typically downloaded automatically and many clients offer the playback functionality in a manner, which renders the text to be the assisting element (vice-versa compared to textual RSS feeds).

Apple has made some extensions to the RSS 2.0 format. These extensions are optional and not required for podcasts to work. First deviation is on the RSS definition field where an argument has been added to declare iTunes name space. Without this declaration, the further tags are to be neglected.

The following lists the additional tags:

- `<itunes:author>` - The element applies to both channels and items and is used to fill the artist field in iTunes (both the player and the music store).
- `<itunes:block>` - Applies both to channel and item elements. Used to prevent an episode or podcast from appearing.
- `<itunes:category>` - Applies to the channel element only. Used to fill in the iTunes Music Stores category column.
- `<itunes:image>` - Sub-element of the channel element. Same location as the album art.
- `<itunes:duration>` - Applies to item elements. Fills in the time column.
- `<itunes:explicit>` - States if the podcasts has explicit content. Usable for both the channel and item elements.
- `<itunes:keywords>` - For both channel and the item elements. Keywords tag offers keywords that can be searched but which are not visible.
- `<itunes:new-feed-url>` - Invisible tag that is set for channel elements. Used to inform iTunes about the new channel URL locations.
- `<itunes:owner>` - Channel sub-element that is invisible to the user. This is used for contact only.
- `<itunes:subtitle>` - Applies to channel and item elements. Fills in the description column.
- `<itunes:summary>` - Available for channel and item elements. This is shown when a “circled I” is clicked in iTunes music player.

Many of the elements are very iTunes and Apple specific and thus their usage may differ on other players. Nonetheless they can be used for added meta-data information about the channels.

## ***2.2 User experience***

### **2.2.1 Words about usability**

Usability is a term used to denote the ease of using an object, whether physical or digital, to reach a goal of the person performing the task. Usability is defined by ISO 9-241-11; “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”. As much as this is a standard, it does not really mean anything on its own.

Usability can be divided into two relevant categories. For one, we have the physical usability and secondly we have the usability of the software. Physical usability deals with interaction devices, design and ergonomics. These are of great importance when it comes to the appeal of the device. However, such factors are left out from this thesis as the primary focus lies on the software layer. There is not that much that could be realistically done with the hardware design given the timeframe, resources and the scope of this paper.

It should be noted that physical usability is in some ways related to software usability, as a study about mobile web design describes (Trewin 2006). This is only natural, as the software is not used directly but via physical interface devices such as keyboards, joysticks, mice and displays.

The study of usability has also lead to various methodologies to measure how well the users can cope with the systems. These can be further divided into qualitative and quantitative methods. Quantitative methods typically measure aspects such as the performance of how the users perform various tasks or ask the test users to rate their opinions with numerical values. Qualitative methods, on the other hand, try to gain insight into the users by seeking answers to questions such as why the users perform the tasks they do in the first place. This data can be obtained for example via interviews, questionnaires or observations. The methods can be used together to provide measurable qualitative results and use the qualitative information to gain insight

It is suggested that there are situations where some other needs compensate the lack of usability and quality of the solution (Reponen et al, 2006). Hence, there is definitely more to the whole satisfaction of the user than from strict usability. Even Jacob Nielsen notes that other aspects make systems such as web pages pleasing even if he does not describe them per se (Nielsen 2000). Thus, the next chapter will deal with a term “user experience”.

### **2.2.2 On user experience**

To create software that is liked by a number of people (by no means all people as that would likely to be impossible), considerations dealing with usability alone are not enough. We need to take into account the holistic user experience, which, unfortunately is rather vague term. The user experience research is still not well understood and not well defined (Roto 2006).

The very definition of “user experience” is controversial. The following is a list of some suggested definitions:

- “Every aspect of the user's interaction with a product, service, or company that make up the user's perceptions of the whole. User experience design as a discipline is concerned with all the elements that together make up that interface, including layout, visual design, text, brand, sound, and interaction. UE (user experience) works to coordinate these elements to allow for the best possible interaction by users.” (UPA 2007)
- “A result of motivated action in a certain context.” (Mäkelä & Fulton Suri 2001)
- “A consequence of a user’s internal state (predispositions, expectations, needs, motivation, mood, etc.), the characteristics of the designed system (e.g. complexity, purpose, usability, functionality, etc.) and the context (or the environment) within which the interaction occurs (e.g. organizational/social setting, meaningfulness of the activity, if the use is voluntary, etc.).” (Hassenzahl & Tractinsky 2006)

- “All the aspects of how people use an interactive product: the way it feels in their hands, how well they understand how it works, how they feel about it while they’re using it, how well it serves their purposes, and how well it fits into the entire context in which they are using it” (Alben 1996)

In essence, user experience means how the user feels about using the product. Naturally this depends on the individual himself as well as his experiences in the past. His location and surroundings may affect how he deals with the task he wants to perform with his device. His expectations also matter. In fact, next to everything is likely to give some degree of “experience”, whether good or bad, to the feeling of the user.

For example, Mäkelä and Fulton Suri (2001) define the user experience to consist of previous experiences and expectations that the user has towards the system he is going to use. The user has a motivation to use the new system and he makes an action by using it in a context (to be understood rather vaguely, as “on lunch break” for example or “finding commuting routes”). Motivation, action and context form the present experience at the time of the use. The present experience then molds the future experiences and the expectations.

A common problem with user experience results thus far has been that the abstraction level is high. It is easy to refer user experience as something that is resulting from the mental state of the end user, his ambitions, lifestyle, location, if it is raining or if he is going to miss his last bus in the evening. This level of abstraction is insufficient for designers to actually help to improve the whole experience systematically and in a formal manner. It does help, however, as a mental model.

Even evaluation of the user experience is not straightforward. It is highly dependant on when and how the evaluation is done. If we ask if a product is pleasing to the user right after he has unpacked it from the shiny covers, he is likely to give a different response than after he has used it a year and the device has just died without a warning.

Virpi Roto (2006) proposes that the term “user experience” would be narrowed down to mean the interaction between the person and a machine. The rest, for example, a person viewing a painting in a gallery, should be called simply “experience”.

While I consider that this definition raises an immediate need to define a more holistic term to replace what “user-experience” used to mean (e.g. everything), this definition of interaction makes the problem area more manageable. While doing so, if one were to hold only to this definition we would omit the possibility to affect this very user-experience with pre- and post-use means (these include marketing and advertising which try to affect the expectations user has towards the product, packaging, taking care of the recycling to name some of the available options). I postulate that a product or software user-experience cannot be done as a black-box; yet, while engineering the software, one is likely to have a limited number of options to affect the rest of the process.

This thesis will use the term in the same scope as Virpi Roto presented in her doctoral thesis (Roto 2006).

### **2.2.3 Notes about emotional communication**

Only about 45% of the feelings and attitude messages shared between people happen by speaking (Mizutani 2006). Actions, such as shaking hands deliver vast amount of information via gestures such as the firmness of the shake and if the partner is smiling or being serious. In fact, Mehrabian proposed a theory that in each face to face communication situation that deals with emotional messages 7% of the message comes from the words themselves, 38% comes from the tone of the words and 55% comes from the body language of the speaker (Mehrabian 1981).

If the above-mentioned messages differ, Mehrabian states that the most powerful ones dominate the message and rule how it is interpreted. Thus, to create a strong message each of the elements need to support each other. It should be emphasized that Mehrabian only stated the relative importance's to apply when the messages

dealt with feelings and attitudes. The rule he proposed was not meant to be generalized for any form of communication.

However, if it is so that in emotional communication only 7% of the message is delivered by the words, it puts the communication via technical means into an odd situation. Via a voice call, we can still hear the tone of the speaker's voice, but when that is taken away, the means to express emotion are relatively thin. Perhaps this has influenced the usage of *emoticons* (often better known as smileys) in text-based mediums such as IRC and instant messaging applications like Microsoft Messenger.

Whether the birth of emoticons was caused by limitations of text based medium or not, the phenomena is documented in context of instant messenger software. It appears that emoticons are used to make sure that messages meant to be humorous are not taken seriously. Emoticons do have a set of problems (such as the duration and scope of the emotion) and thus methods to expand the emotional communication in those mediums have been attempted (Sánchez 2005). Some methods have even been offered to tackle transferring the gestural messages within text (Adesemowo 2005).

Emotional factors also lead to the subject of massively multiplayer on-line games (MMOs). These types of games rely heavily on other users and the interactions between people to make the player belong into the world. The game itself can often be mastered relatively quickly and offers little what is not available on stand-alone games often played on computers and game consoles. The real hook is to provide a social context for the players to stay with the game and its subscription even after the initial objectives have been reached.

Some have, however, stated that the importance of these social networks is overstated. The obtained results have not indicated that the social aspects do not matter, though. Instead the games such as "World of Warcraft" benefit from merging the games to the community activity, offering the players both at least to some extent (Ducheneaut 2006). Others have also noted that the players with heavy interactions with other gamers tend to play the games longer (Chen 2006).

Each of the examples emphasizes the importance of human factors and gives hints that the emotional bond between individuals would affect their feelings of a device or an application; either directly or indirectly by providing them the connection they could not get otherwise. In the case of MMOs, choosing a new one would require the player to rebuild his social network again (unless, of course, the existing network would move with him).

Thus we are left with an interesting question: “How can we facilitate the people to create a bond to either the software itself or to other people reached with this software?” Michihito Mizutani proposes a “French bulldog theory” in his thesis:

“Suppose that you have a French bulldog at home. He is really lovely, but none of your friends understand that. One day while walking your dog in a park, you come across another French bulldog. You excitedly start a conversation with its owner even if you have never met them before.”

The idea he presents is not too surprising and within dog owners quite well known for some time. Another example he mentions is a “Sole Bag” designed by Naoto Fukusawa. Again, the actual product value is experienced by the user and easily omitted by the casual observer.



*Figure 2, The “Sole Bag” by Naoto Fukusawa. The bottom of the bag is made to resemble the sole of a shoe. This is designed to ease the mind of the users as they can now lay it down to the ground without being afraid of dirt.*

The situations he describes come from the object facilitating the discussion and the creation of a new social contact. The individuals have something in common, a peer group of sorts. A product can also be a tie to a specific social group an individual wants to belong to or to be identified by. Such trends can be seen in fashion where the clothes are a method to express oneself.

In fact, a similar phenomenon is not uncommon in software and digital world either. There are people who are acknowledged supporters of various applications, such as operating systems. The common interest has caused people to create communities (such as Mac and Linux supporters) and made people feel strongly for their chosen software.

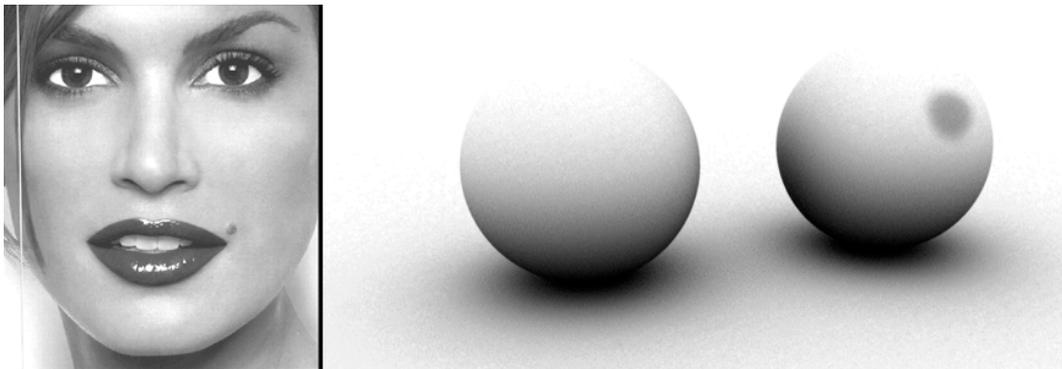
#### **2.2.4 Theory of broken perfection**

While the messages that pass between people are undoubtedly important to create an emotional contact, there are other aspects as well that affect how we perceive things. If we consider the “French bulldog” theory from the perspective of the initial owner of the dog, the social needs that are fulfilled by owning such a dog are not likely to be the ruling factors (they can be, on occasions) when you choose which kind of a dog you would like to have. After you have made the decision to get the dog, the emotional connection to other dog owners comes naturally. But why would one get a French bulldog in the first place?

Some objects have appeal to people because they stand out from the rest. This is particularly notable in regards to fashion industry, but is no limited to it. Some software have also created an avid audience by (or, one could argue, despite) providing unique interfaces that stand out from the norm. Blender could be considered one that differs greatly from the dominating 3D software line-up (e.g. Autodesk Maya, Avid XSI and SideEffects Houdini). Another example is Pixologic ZBrush, which too introduces a new interface style for sculpting 3D surfaces (in contrast to Skymatters Mudbox and Nevercenters Silo, for instance).

If one designs what he wants in detail, there is a good chance one might end up with something that he does not like in the long run (assuming that the designer is the user as well). Knowing every little detail and being able to predict the future behavior of an object can take part of the fun out of it. In fact, aleatory filmmaking has been experimented by directors such as Fred Camper with the movie “SN” and Barry Salt with “Permutations”. These avant-garde films took the chance of randomness to the very experience of their movies.

For example, Cindy Crawford has a mole next to her mouth. That mole breaks the symmetry, or “perfection”, and makes her face more interesting. Similar oddities are seen in the placement of the Saabs (Swedish automobile manufacturer) keys; they are located next to the hand brake and not in the steering wheel like most other car manufacturers have decided to do. Mac OS X introduced the small window closing, minimizing and resizing icons that are on the “wrong” side of the window; on the left while on most other window managers they reside on the right. All these little quirks make the things different and often more attractive. It could even be thought that such personalities together create the very soul of a product.



*Figure 3. Cindy Crawford (on left) has a mole on the upper-left side of her mouth breaking the symmetry. In similar manner, the rightmost rendered ball is more interesting than the one on the left because of the detail that draws our eyes to look at it longer.*

Together with the appeal to a certain set of people comes the risk of alienating a large (or even largest) group of users. There is no denying that a good percentage

of the population is rather conservative and feels intimidated when things don't work exactly as they have used to. In general, people have a tendency towards resisting change, or preference towards familiar systems (Butler 1996). Such behavior, however, should not be taken too strictly. When cars emerged, people were rather suspicious towards them as well. For example, there were strict speed limits and a person was required to walk in front of the car with a red flag to warn others of the vehicle approaching them. If we do not challenge people to think differently, little progress can be made. Still it needs to be remembered that there is a fine line and possible penalty to pay when walking your own roads too much.

To emphasize; this theory does not mean that usability should be neglected nor does it mean that usability has no value. Contrary, the usability has immense value for the end user, and this theory only claims that it is not the single aspect end-users care about. Quirks should not contradict usability and traditions too much; little might be allowed, but if they destroy the users' ability to use the device or program there is little benefit from being different for the mere sake of it.

### **2.2.5 Emotional value chain**

In the previously noted "French bulldog" theory, the communication is in fact occurring between the people. The object that bonds them and starts the conversations, merely presents their likings and signals they may have something in common. The emotional value, the message exchange with the other person, can be seen from moving with the help of the facilitating objects.

Similarly, if we receive a photograph that portrays a beloved family member, the significance is not in the physical photograph. In this case, the value of the image is the emotional contact we have with the persons in it. Again the object merely transfers the emotions, possibly over distance and time.

The emotional value chain is a term used in the scope of the thesis to describe this facilitating nature of objects or elements, which have similar properties.

## ***2.3 State of the art***

### **2.3.1 iPod, iTunes and iTunes Music Store**

One of the most well known ways to listen to podcasts is the combination of iTunes and iPods. iTunes, a media player software that runs on Windows and OS X operating systems, serves as a directory for individual casts. The podcasts are combined with the iTunes Music Store (iTMS) and the interface provides ways to both search the content and see what is popular. The popularity aspect has drawn some criticism, but nonetheless it does give an alternative method to find possibly interesting podcasts.

As a result, podcasting in regards to iPod and iTunes needs to be considered as an ecosystem and that is what separates iTunes from many competing systems. Each of the components does their part and works together. Considering that iTunes runs on a personal computer, it can take advantage of the Internet connection the PC might already have. Thus an iPod does not need wireless connectivity or even connectivity to elsewhere but to the host computer. The second benefit comes from the power consumption; typical desktop computers have a power cord and thus do not need to rely on batteries while they are operational. This allows more aggressive updating of the podcast subscriptions and downloads.

On the user experience and usability level the combination also provides different interfaces for managing and consuming the content. The mobile part does not try to do everything, but the task it does do (sorting, finding and listening the already downloaded podcasts), it does very well. The interface consists of a disc that has five buttons and the possibility for the user to move his finger on the disc to perform further actions such as scrolling lists and adjusting volume, for example.



Figure 4. The front page of the iTunes Music Store is accessible with iTunes media player and offers various ways to browse through content the user can subscribe to.

Also, the task of finding podcasts is not trivial from the interface point of view. This too is taken away from the iPod and included into the desktop software instead. The desktops typically offer much more screen estate and have more flexible input methods compared to most mobile devices, typically the combination of a mouse and a keyboard even if more exotic ones can be used on circumstances (such as tablets).

### 2.3.2 Nokia Podcasting Application

The Nokia Podcasting Application is designed to work strictly in the mobile context. It does not have tight connections to other devices or to podcasting directories outside the device itself. The application runs on S60 platform and uses over the air connections to download the content to the memory card on the device.

The program starts with a menu for the user to select suitable actions, which include a directory hierarchy to choose the shows to subscribe. This directory is

formed via OPML (Outline Processor Markup Language) files and updated when accessed. After the subscription, the podcasts are moved to a user-managed hierarchy that can be organized via folders.

The downloading of the podcasts happens based on free memory requirements and the time interval from the last check. If there is no sufficient free space, downloads are postponed. Similarly, the checking of new downloads only occurs after the interval and thus the notifications about the new content is not immediate.

The Nokia Podcasting Application does not contain a server module. This requires the application to be running on the device for the whole length of the download and that the device has network coverage and connectivity for the duration of the download.

While the application does not have contacts outside the mobile device, it does benefit from other existing software on the device, namely the media player that is used for playback. The integration happens when the user has selected a downloaded podcast for playback. Instead of launching the player of its own, the Nokia Podcasting Application starts the platform default player with the audio file.

What makes Nokia Podcasting Application interesting is that it operates in the same environment as the client that was created for the purposes of this thesis. This makes the evaluation easier as comparisons are not dependant on external software and arrangements, meaning that the only differentiating factor is the software itself. It is quite probable that the other factors do affect the overall user experience a great deal but in the scope of the thesis factors such as hardware design were out of the scope.

### **2.3.3 Other implementations and related work**

Podcasting is an adaptive use of RSS feeds, and thus the underlying technology has been utilized on textual medium even before podcasts became the phenomena

they currently are. Due to the history, many RSS readers deal with similar concepts of subscription, automatic updates and browsing through the content.

There are many RSS readers, however. Nonetheless the basic interface seems to have become relatively standardized to contain three major interface elements. These are the list that shows the user defined subscriptions, a list of news items in the selected subscription and a preview pane where the content of the news item can be read. Such a setup is in use at least in NetNewsWire (in both versions, including the Lite version), Omea Reader, Awasu, Firefox (with Sage extension) and RSS Owl. The Apple Safari web browser has actually has a slightly different methodology where it shows the item previews with the items themselves. While this list is not exhaustive nor necessarily even generally representative it does contain notable RSS readers on two different platforms (OS X and Windows). Most of the client applications are also configurable, so the basic layout may not be what the people end up using on their own computers.

RSS and podcasts have many other similarities considering the method of how subscriptions occur. While the readers may contain directories, the emphasis is on finding the interesting news from web pages. It would be speculation to say that the web-based subscription is the most common method, but at least there are not many well-known RSS directories on the web.

Many of the RSS applications also have a support for enclosures, making them effectively podcast applications. The separation could be considered to be in the automatic functionality and tight bundling of the media playback capabilities to the host application and a way to transfer the audio to portable devices. In fact, such software is needed to gain iTunes-like functionality with portable players that are not supported by iTunes (natively or via plug-ins) or with a device that needs to operate under various operating systems. As with RSS readers, these applications are not rare.

A somewhat different use of the very same technology is available in the “Democracy: Internet TV” application. The software can be used to subscribe to RSS feeds via self-contained directory much like iTunes Music Store, but also

offers search possibilities. To be able to search audio-visual content, it contacts other services such as YouTube, Yahoo and Google video to perform a keyword search against the tags in their content. The results are then presented to users and can be saved as an updating, virtual RSS feed (the application itself uses term channel for the operation).

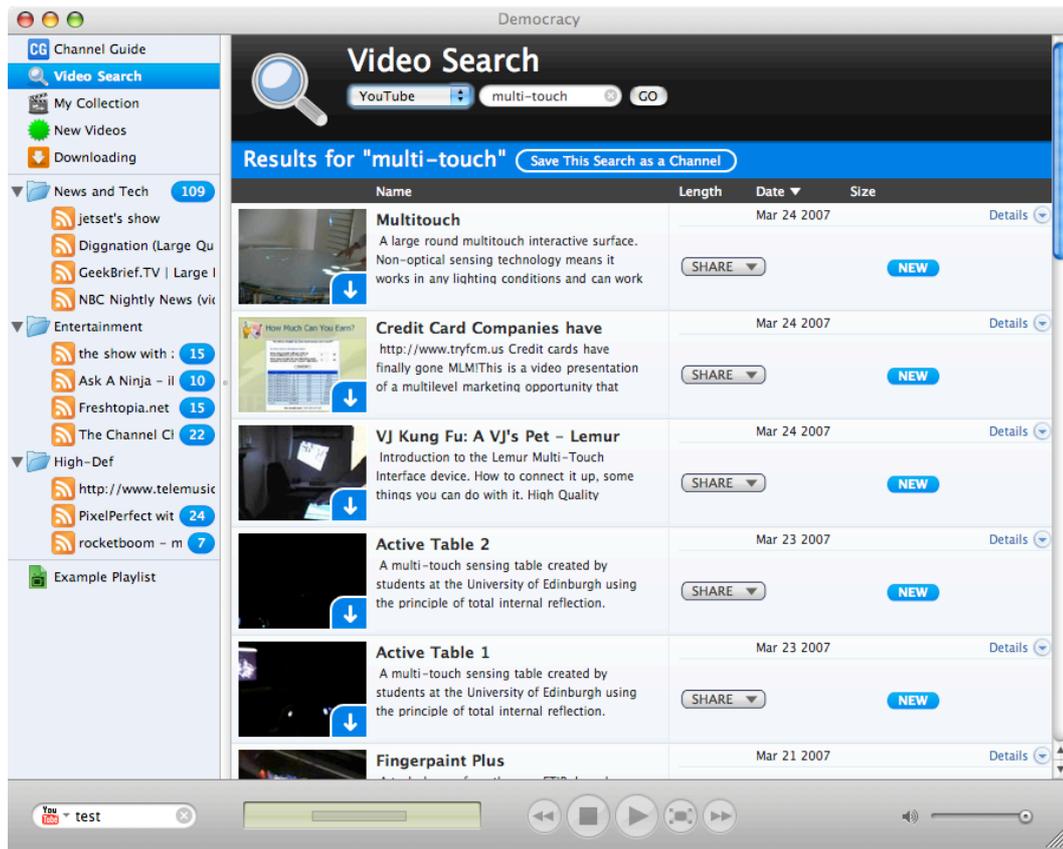


Figure 5. Screenshot of Democracy: Internet TV application that makes heavy use of RSS to deliver updating video content and allows searching through different services.

## **3 Podcast application**

### ***3.1 General design issues***

Designing systems that are efficient and easy to use has not been an easy problem to solve. The problem has been tackled in multiple ways throughout recent history and with varying success. The early problems mostly dealt with efficiency and perhaps one of the most well-known design results is the QWERTY keyboard layout that was created for typewriters. The letters in the early machines jammed easily and thus the keys were re-arranged so that the most often used letters were as far from each other as possible. This, in turn, made it less probable for the rods that hit the ink tape to get stuck with each other and allowed the typists to type faster (Milne 1998).

Software design has gone through various methodologies as well. Early on, personal computers were limited in what they could do with reasonable computational and programming effort. Simultaneously the software development was a relatively young profession, and it should be noted that personal computers have existed for only a relatively short period of time and the history of non-mechanical computers is not very long either.

One of the approaches taken has been called system design and it is mentioned here as an example which can be considered as an almost opposite methodology to user centered design (in practice these methodologies need not to be mutually exclusive, however). In system design, the general approach has been to identify the system requirements and to design software that fulfils those targets. In a traditional sense those targets are set so that the system technically works but may or may not deal with the concepts how the information is organized and presented to the actual users on interface level.

At some point it was noted that fulfilling the technical needs was not enough. Users could not use the software easily despite the fact that it worked as specified.

There were many causes for this. Sometimes the terminology used by the software was different from the one the people used themselves. At times the most commonly used features were hidden under deep menu structures where the less used functions occupied screen estate. The interface might not have offered affordances and would require heavy use of a thick manual. Many books are written on the subject, such as “Emotional Design: Why We Love (or Hate) Everyday Things”, “The Design of Everyday Things” (both by Donald A. Norman) and “Usability Engineering” (by Jakob Nielsen).

The situation eventually led to a methodology that is known as “user centered design”, though this is not to say that it is a better than system design approach. In user centered design the process (or philosophy) is driven from the perspective of the user. User centered design emphasizes the involvement of the users early on in the design and evaluation phases of products (UPA 2006) and thus the product is not defined by a list of technical requirements alone. However, many methods that fall under the umbrella term “user centered design” rely on involving the users in the actual design of the product.

Some of the methodologies that rely on user input are listed here; participatory design, contextual design, co-operative design to name a few. Each of them is at least a semi-formal methodology that can be followed to help achieve a pleasing design.

The difference between system design and user-centered design is not as black and white as it might seem from the earlier description. Usability factors can be taken in account even if the system is made from technical requirements and the interface designs might well be separate component designed independently from the rest of the system. It should also be noted that no methodology developed thus far guarantees a pleasing result. In my opinion the major benefit gained from the formal methodologies is that they make it easier to avoid pitfalls that affect the user experience negatively.

### ***3.2 Design drivers***

Designing software is rarely easy and it is nearly impossible to please all of the audience. People have different needs for the same software and not only do those needs differ from a person to another but they can do so even in the context of a single individual.

The users may even have different roles, or identities, based on where, when and why they use the software. A simple example of this is work and home identities, which can differ quite a lot from each other. In a work context a person might look for information from Google in regards to pneumatic pumps, but when at home he would be more interested in Dilbert (a comic strip <http://www.dilbert.com>).

As the products are mostly designed to fulfill a purpose (whatever that purpose might be) and often one of the goals in development is user satisfaction, the solutions need to rely on decisions. These decisions are derived from what is called design drivers.

Design drives are a set of assumptions and motivations that can be used as a tool to make decisions. They usually describe target audience and set goals that the design should fulfill. Some of these can be political in nature (e.g. “we wish to encourage everybody to use Internet”) whereas some may be set by the market research (“it seems that there is a demand for a service that does thing x”) or the user segment to which the product is to be targeted (“people who go to work by bus”). Factors such as estimated price (for production or the street price) and manufacturing can set their own limitations that may end up affecting the design.

The eventual designs can be evaluated later against the design drivers by studying if the outcome can be used in the role it was specified. Evaluations can be done with the representatives of the defined target audience, which gives an indication if the solution is right for their specific usage.

### **3.2.1 Role of the application**

The key issue for this prototype was to emphasize the messages delivered via the podcasts. The message, in fact, is that of the content (audio and attached visual information, including explanatory texts) and unfortunately it is impractical to assume we could affect that as such.

However, such would fight against the long-tail assumption from both the side of content producers and on that of the consumers. Producers have to learn and master a challenging set of skills to produce content that delivers their message in the way they intend and tampering with that content would need a very good reason. Also, the audience might expect to hear the message they subscribed to and not the altered version made by the software in-between. Examples of such alterations would be, for example, changing the album art, censoring the informative texts or even the audio tracks and optimizing the audio with filters and re-encoding for bringing out the speakers vice from background noise. In case of video, we might think of enlarging parts of the image to highlight what our algorithm thinks as essential in hopes to make it better stand out on small screen. Also, it would be naive to imagine that our preferences would apply to everybody else. Thus, tampering with the content or producing the content itself (instead of the current publishers) would not be in any way feasible or desirable.

This leaves us with the software and interface alone (as even the hardware design of the mobile devices is out of the scope of this thesis). The software should fade away and leave the stage for the content and only underline what is already in the podcast. After all, the user selectes what he wants to listen to. Jakob Nielsen states similar ideas in his book “Designing Web Usability” (Nielsen 2000). Speaking specifically about web pages he states that the emphasis is to be put on the content instead of various other aspects such as site navigation.

The application interface needs not to be indifferent, however. It should be responsive and concentrate on providing the content quickly and easily. It should also make the needed features pleasurable to use.

### **3.2.2 Targeted user group**

As podcasting is a relative new and growing phenomenon and has yet to fully mature, the emphasis of this study was to make the consumption of podcasting more appealing to a non-technical audience. In practice, this means the people who may not have even heard the term “podcasting” before but still might enjoy the content without delving themselves too deep into the details how the whole system actually works.

The preferred age group would be from the young adults to those in their near-thirties. It is quite likely that the actual audience varies much more, but this age distribution was assumed to be potentially the most interesting one to take the new delivery mechanism into use. The gender of the audience was not considered to be a factor.

It was also very clear that requirements to use the prototype caused even further segmentation in the user group. For the prototype to work it was needed to run on a fairly modern S60 platform. The reasons for this were mainly practical ones: the availability of hardware and the technical resources and feasibility of the development. There is no particular reason why the same concepts and even interface would not apply to other platforms as well.

The amount of downloaded content might limit the audience as well, depending on their contracts with the operators as well as their access to wireless networks. At the time being, the cost factor is likely one the most limiting factor for people not using data traffic more than they do now (Halvey 2006).

People with hearing disabilities were not to be taken into account in the design due to the nature of content itself. While the task is likely not impossible, to tackle it would likely require a new project as extensive as the current one to deal with techniques such as audio-to-text translations. In milder cases this could come down to altering the sound characteristics of the feeds to help people to cope with less-than-perfect recordings.

If the hearing was not severely limited, many would be able to increase the sound level and use headphones to overcome the troubles of hearing what is being said.

The visually impaired were not accommodated in the design to any great extent either. Effort was put to make the text and descriptions readable, but the interactions with the hardware and software were expected to be done via available buttons and the joystick instead of more specialized methods such as Braille terminals. While technologies such as speech recognition and text-to-speech were discussed, no implementation effort was made to incorporate them.

It should be noted, however, that the ability to use the media player without watching the screen was deemed important. This should, at least on its small part, lessen the need to pay attention to the visual elements, as the responses from the software would be predictable. Benefits such as this appeal to many audiences and help in situations where the user is moving and his attention is more focused on the events that happen around him (e.g. while walking in urban environment).

### **3.2.3 Appearing simple**

While it could be debated whether an application should be simple or not, one of the basic assumptions behind this work was to make the podcast client to at least appear as such. The emphasis was especially on the long-term usage of the software.

The means to reach this aim were to use fewer key presses than previously, not to resort to lengthy options menus as done commonly in the S60 environment and to initially start with a small set of crucial functions from which to expand later on. Also, features that the user does not really need to care for were to be put away from places where they otherwise would get the immediate attention.

The paradigm of “simple use” ought not to be mixed with the concept of having fewer features; due to the scope of the application many features present on other clients would likely to be needed, but that was not a driving motivation as such. The user interface was to hide unnecessary details from the end users, such as

byte size of the download when the more relevant information could be, for example, the length of the episode in minutes and seconds.

As it is difficult to know what the users expect from the end product and what the actual, relevant information that the users expect is, this work tries to offer a sensible guess and see how close of a match it will be with the feedback from the users. Such guesses were also to be tested in the user trials.

### **3.2.4 Assisting features**

As the main motivation for the users should be to access the content, and not to use the developed application, the ease of use should be specifically considered and help provided where possible.

As many as possible of these assisting features should not interrupt with the flow of the software. In practice, this would mean that the users would not need to stop to wonder what happened and that some questions would be clarified even before the need to ask about them would arise.

Since it was likely that the interface would use animated elements, special attention was put to describe the requirements for such animations. Animations happen over time, and thus require a beginning and an end separated by some finite amount of time. This time is by its nature time taken away from doing something else that could take place instead. Users are put to wait for the software, even if the time interval is minimal. Negative effect can be seen, for example, on interfaces where animations are used on list controls. The highlight is moved to next or previous one with a smooth animation as can be experienced, for example, with some parts of the Nokia Channels (Kanavat) application version 1.32.

Thus, animations should be relatively quick. A common problem with animation in user interfaces is that they hinder the usability by being too slow and making the user to wait as he is doing repetitive tasks. In these cases, such as list controls, the use of animation ought to be minimized.

### 3.2.5 Assumed use-cases

The main benefit of a mobile terminal is the fact that it is designed to be mobile. It moves with the user. It is available most of the times when he needs it if not hindered by social (social pressure from other people near by, the situation and context of use) or technical matters (e.g. the device may not have enough battery power, safety regulations forbid its use). Thus it is natural that the actual usage of the podcast client is to happen on the go, either while moving or waiting. The latter is in fact a rather common situation as observed in Nokia internal usability tests. Being mobile also means the user must be able to control volume according to his surroundings.

While subscribing and handling the removals of podcasts or podcast episodes are both important, these should not be the focus of the client. This thesis is mostly interested in the consumption aspects. The hypothesis is that the users will subscribe to a set of podcasts and listen them as new content arrives. Most of their time ought not to be spent micro-managing the subscriptions themselves.

In this case consumption means the listening to a set of episodes or possibly only one specific episode. It can include the viewing of possible album art and information associated with each episode (if either exists in the first place), but as the content is mainly in audio, the act of listening is the deciding factor when deciding if something has been consumed or not.

It is also assumed that listening happens with headphones. Given the social environment in Finland and the emphasis on the scenarios where users are moving on populated areas, it seems unlikely that usage of the built-in loudspeakers would be common.

As a result of these assumptions, special attention is paid to the scenario where the device is out of the visible range of the user, for example in the pocket. The controls should work without the visual input once the player has been started and the first episode selected. While this is not possible for all of the functions, at least the playback should take this into consideration.

### **3.2.6 Product life-cycle**

Products do not come from nowhere nor do they last forever. The result is that the products typically have a life cycle that is shorter than their owners and thus this chapter outlines the simplistic cases the product goes through in the hands of its user. While many of these stages deal with aspects that are out of reach for a single application developer to affect, they must be analyzed in order to understand the user and the implications the chosen courses of action may cause.

For the user to start using a product, he must at first get it from somewhere. This can happen via a retail channel, but the actual methods may vary greatly. As a result, the user must somehow know about the product and something must have caused interest in the user.

The podcasting client is, ideally, pre-installed software on a product. The actual product, a mobile handset, is not assumed to be bought because of this single application but for the other benefits it provides to its user. Thus one obstacle to the adoption of the application is circumvented and the likelihood that users start to take advantage of the application is increased, as there is no separate process of downloading and installing the application.

After the user finds the application, he may actually launch it for the very first time. This also defines the initial experience and communicates further with the user what the application is used for. This may not be self-evident from the name of the application or even from the possible download pages that the user has gone through (in case he would have needed to install it himself), even if on this occasion he is likely to have some idea why he got the application in the first place. But for an application that is installed on the platform by default, special attention is needed to make it clear what the application is and what it is used for.

More particularly in the case of feed readers and podcast clients, a common approach is to have some pre-determined content, which the user can browse from the very beginning. The content may not be to the users liking, but at least it is likely to give indication of the proposed usage. Some software products also use

introductory screens that tell about the program and introduce the features to the users (including Adobe Photoshop, for example).

As the user starts to use the software more often, the situation changes considerably. The instructions how to use the software is needed less as they learn from the previous times that they have used it (hopefully, at least). In fact, what might have been a beneficial feature early on in their usage may now become a usability issue. In the case of a welcoming screen, the advanced user might already remember what it says but he still is required to go through it or inactivate it manually (again, as is done with Adobe Photoshop).

Also, as time goes by, it is to be hoped that the users select content that they personally like, and thus they can form an understanding of what will be on the device. They can even anticipate when the updates arrive. Assisting features such as animations help less and can become an obstacle, as the user knows where he is going to and tries to get there quickly.

In the end, the user will stop using the product and the application. It may be that he has lost the interest to continue the software or that the application no longer meets his needs due to a change in his life. Again, the reasons to abandon a product are many, but eventually such a moment arrives. What can be done is to take the positive alternatives into account and make the parting from the product as pleasant as possible.

In the scope of podcasting client, this could mean that the move to a new version or even alternative application is made easy; the user subscriptions are transferred to a new device without hassle together with the possible downloaded episodes and information of what he has and has not listened to.

## 4 Implementation

### 4.1 *Initial prototypes*

#### 4.1.1 Initial feed-reader enhanced to podcasts

The very first attempt to create a podcasting application took place during the second half of 2005. The design was based on an existing RSS client and this was enhanced to deal with audio enclosures. Also, support for subscription icons (much like *favicons* on web pages) was included.

The implementation then went to user trials in December 2005 and several usability issues were identified. This prototype used many animations, and it was commented that they were not pleasing and offered very little value to actual usage. Also, as the prototype was included to the web browser the need arose to have a way to quickly navigate to the audio content. Offering a screen from which to select between RSS, Podcasts and the web browser solved this problem, but the test users did not praise it.

The issues lead to the conclusion that it might be better to keep the applications separate and thus a new prototype in 2006 was created as a separate application that could be accessed directly. Application was also set to take an advantage of hardware 3D acceleration to solve the issues encountered with animations. With the previous 2D engine the speed could not be affected too much and the requirements for the explanatory functions were harder to overcome in the old environment. An added benefit was that the implementation did not need to care about the complexities of the other functions in the web browser.

#### 4.1.2 Paper prototypes & interaction diagrams

One of the methods to get early feedback is to have the designs evaluated with paper prototypes. The states, or views, of the application being developed are drawn to the paper and the uses-cases are gone through with the test-users. Paper prototypes are often used as a way to communicate with end-users and to emphasize that what they see is a draft and that making changes to the views is

easy. This is said to make the commenting more relaxed and informal (Grady 2000) as the users would express their opinions about the prototype more easily. Seeing that changes to paper prototypes would be easy to make, users would not feel bad about suggesting changes to the existing solutions done by the developers. Also, being able to make rapid changes without a new round of evaluations can speed up the development.

It has, however, been noted that paper prototypes may have a negative effect as well (Lin 2006). Given the nature of possibly unpolished drawings, some people may not take the situation seriously enough to give as valuable comments as they otherwise might do. Also, some users seem to have difficulties evaluating the draft level paper prototypes (Rudd 1996). The attention may drift to details that are inessential to the prototype at hands. Further, when the paper prototypes are produced by hand, the screen estate is no longer given and specifics such as the size of the writing may cause serious problems later on when the drafted designs are modified to fit the screen.

In this instance, the paper prototypes were not tested as described above. Instead an interaction diagram was produced. This diagram visualized the interaction between the user and the software by connecting the key presses to the state changes within the application. Such a diagram helps to make sure that there are no lapses in the planned interaction and that the navigation remains logical and consistent. Further benefits include the easiness of explaining the design to others; thus it serves as a communication tool. However, this may not be a suitable tool to be presented to casual test users.

The actual interaction diagram was made via printed screenshots that resembled the drafted looks of the planned application. The reasons for this were many. First, it does give more concrete feedback about what is possible to fit on the screen and remain understandable. The drafting with an image-editing program was also efficient given the authors' skills with the selected program. And lastly, usability experts who were used to give feedback on prototypes even with more polished interfaces evaluated the draft.

### **4.1.3 Expert evaluations and changes to the final design**

The interaction diagram was presented to a selected set of usability experts and developers by taking printouts of the screenshots and collecting them to a large sheet of paper. Eventually the pieces were connected to each other by coloured marker pens to show the results of the key presses in each state of the software. The flow of the application was verbally explained and the evaluators could ask details where needed, even to the extent to follow through their own attempted use of the software. In fact, almost the full interface functionality could be tested with the exception of the pieces that required outbound connections (such as visualizing a podcast directory that was left out of the scope of the client design).

The outcome of the session had the greatest effect on the design of the player, and uncovered the importance to show information as early as possible. Suggestions also dealt with the type of information present on each individual screen. Modifications were made to the actual prototype implementation where possible and when considered sensible. Also, some suggestions were taken partially, as is the case of selection lists that incorporated a zooming functionality. The original suggestion was to enlarge the selected list elements and show additional information with the extra space they offered. This might have happened by showing an additional row of text under the element label. However, the prototype implements only the zooming to help the readability and emphasize the current selection.

## ***4.2 Draft implementation***

### **4.2.1 Color scheme development**

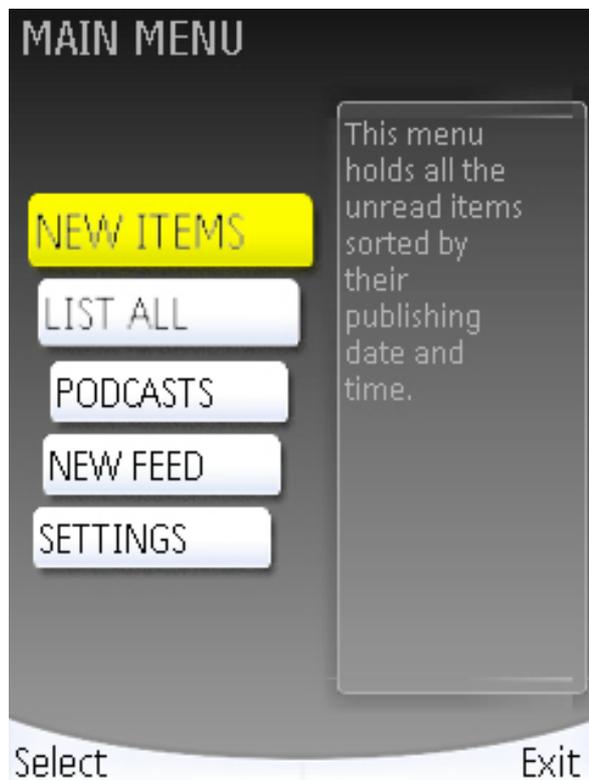
The size and limitations of the target audience were considered while implementing the color scheme for the application. As the target audience was to be considered relatively young (at or below their near-thirties) a less conservative approach was taken.

The first attempt visualized the application as rounded black buttons with yellow rims. Selection indicators were done with a glow around the buttons. While the

general feedback was relatively good in regards of the looks, the pleasantness divided the peer opinions. The opinions were gathered from a group of 6 people to test their initial reactions (of which only two were females).

As a result the “techy” look was toned down to some extent and the selection highlight turned into a yellow color overlay. The dark background was made clearer and more polished by removing the noisiness of the original gradient.

The peer review of the changes was encouraging and thus the overall look remained to the actual user trials. The trials were expected to reveal the general acceptance of the user interface and see how the color scheme affected the users opinions.



*Figure 6 presents a screenshot from the eventual implementation. The selection list has become more rectangular from the previous design while the smoothness in the options bar at the bottom reminds from the origin. Some of the elements of the interface (such as font rendering) depend highly on the underlying operating system version and thus can vary from one device to another.*

### 4.2.2 The “Wow!” effect

At the time of writing this thesis, the so-called “Wow!” effect is rather easy to implement due to the immaturity of the software culture and the lack of general understanding of story telling among the software makers. Little by little this is changing and there starts to be more weight on experience aspects of the software. An example of this is seen with the AJAX (Asynchronous Javascript And XML) assisted web pages that concentrate on usability and, to quite some extend, to be flashy.

Aristotles is often mentioned as the formalizer of the western story telling tradition by describing the story arc that is seen in many of the concurrent pieces, including the recent Hollywood movies as well as the works of William Shakespeare. The basic principle is that the story starts with an introduction that soon arrives to a conflict, which tries to hook the users interest to the story. The story then continues to increase the intensity towards the climax near the end where the problem is solved. After that, the story should be gracefully ended to give satisfaction to the viewer. The formalization of this has been criticized (Egri 1972), but it is still the prevailing theory in western story telling culture.

The “Wow!” effect should have its basis in the very same structure; the software is first introduced to the user and after a while, there comes a (positive) surprise that solves a problem in the usage in a unique way. The exact method needs not to be graphical. Some differences to Aristotles model about the story arc are that software usage does not have a similar ending and do not necessarily begin in the same fashion. The usage of the software might have a reason, the initial problem, but often the case is that this problem to which the software is the solution does not end after the set time. There are exceptions to this, such as adventure games that typically have a limited life span.

However, there is a possible conflict with the “Wow!” effect and the long-term usage of a given application. The impressive features get easily counter-usable and start to irritate users and thus their usage should be kept where they actually make sense and ease the users job to do the task he planned to do with the given

device. The “Wow!” should not come to his way and prohibit what he plans to do. A positive example of this is Apple’s Exposé that eases window switching on desktops by arranging miniature images of open windows from which the user can choose the one he prefers. The effect is created fast and it gives a visual feedback and a memory trace to help the task of picking the right window. Thus the functionality is more than merely pretty and surprising, it possibly gives a long-term usability benefit at least to a set of users.

The “Wow!” effect in this work was not planned specifically, but as a side product to the other functions. Special care was made that each time the user changes a view or moves inside a menu hierarchy, the effects are animated and tried to make look both appealing and fast enough (in this particular case the animations latest 300 ms.) not to bother even in long-term usage. The animations emphasized the information, where possible. For example, once the user selects to enter a sub-menu, the animation moves the entire lists to center the new one to the screen on the basis of the location of the item in the first list. When moving back, a similar animation tries to give hints if the user is in the middle of a long list, in the beginning or at the end of it.

The media player view implements the default screen in a way that shows the album art in very large size compared to the overall screen. The appearance and disappearance of the image are also animated.

### **4.2.3 Emotional functions**

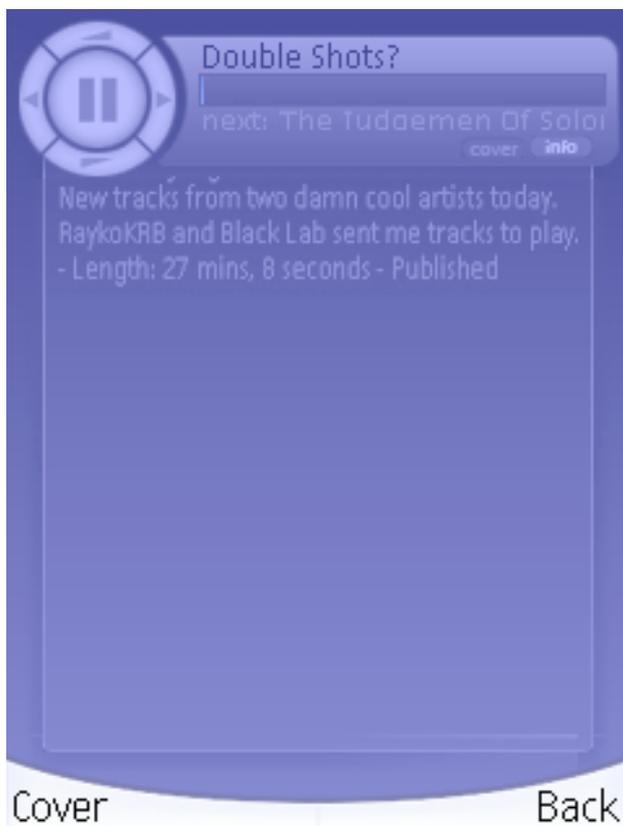
As stated in the design drivers and the chapters dealing with the emotional transfer between actors (most typically users of the software, but not necessarily) the podcast client was to stay in the background and only emphasize the message that was passed from the publisher to the listener.

This message contains three different audiovisual parts: sound, text and imagery. The only one that can be counted to exist is the audio component. Album art, the image, seems to be increasingly common based on the small set of examined podcasts and is also a strong element as it identifies the podcast either on the episode or subscription level. It can tell by its graphical presentation a lot about

the content if prepared carefully. In a way, it can be considered to be an advertisement for the show and offer a possibility to the listener to identify himself with the author.

Thus album art serves two purposes in the context of the prototype. First, it is shown in large size and is briefly displayed to offer a “feeling”. Secondly, it is displayed on the screen by default whereas the textual information is put behind an extra key press.

The player also starts pulsing with blue color while the audio is playing. This serves no actual practical purpose, but is intended to create a connection between the person listening the podcast and the device when the listener happens to look at his device. The pulsing hopefully facilitates an emotional bond to the person authoring the podcast by making the device that delivers the message more alive while the sound is played back. As added benefit, it also visualizes when the application is actually playing and if the silence is part of the show or not.



*Figure 7. The player pulses in blue when the podcasts are playing. Motivation for this is to create a contact between the listener and the sound he hears through the loudspeakers if he happens to look at his device. The effect has been created by adjusting opacity of a blue canvas drawn over the actual interface.*

#### **4.2.4 Applying the theory about emotional value chain**

The proposed theory about emotional value chain deals with the emotional connection between people, it is by its very nature is dependant on social aspects. As such, there is a limit on how far a technology can go alone; there is no known replacement for human-to-human aspects with the current technology. Thus, that is the very reason why the very aspect needs to be emphasized to create an application with more emotional value to its users. Simply put, a podcasting application is unlikely to create the same feeling of attachment to the listener than the person creating the podcast.

To some extent, the emotional attachment can be enhanced by sharing as much detail as possible about the person authoring the podcast. In many IM (instant messenger) clients one can use both a screen name (a name, or alias, that they have set for themselves as an identifier and needs not to be their own, real, name) and an icon or a photo about oneself to be identified by. For an IM messenger, these are a natural subset (more info can be added, such as descriptions about oneself, work, hobbies etc) but do not relate to podcasts all that well.

With Apple's additions to the RSS standard the possibilities for this meta-information have expanded to some extent. There is a possibility to add album art to each podcast and to each episode. It is also possible to share textual information about the feed and the episodes. Recently the options to use indexes within the podcasts, with each subsection having its own images, were introduced. In addition, there is metadata such as the sizes, dates and enclosures, but these rarely include emotionally relevant information. There is, however, no reason why they could not.

The most important piece of information about these was assumed to be the image, the album art of the current show. For that reason its role was emphasized. This has the drawback that there is no enforcing about including such images and

there is a chance that they are not present. In these cases, the client should default to the next emotionally important piece of information.

The textual information was assumed to take this role, even if its existence would likely depend on the podcaster. As the technology matures, we can hope that more and more effort is put to the descriptive texts, by the author, and thus the usefulness of this information would increase.

The last information, again from the emotional point of view, is the technical data about the length and the size of the podcast. Neither of these have direct emotional function, even if such can be delivered to some extent by assuming information about this data. The length can tell if the episode differs from the other episodes and if it does, there can be something special in it. From the size of the enclosure one might be able to deduct this length.

All of these sets of information have usability purposes as well. Album art serves as a quick visual identifier, assuming of course that the image is familiar to the user. Textual information may be used by the user to get a brief glance if he should reserve time for the episode, push it back later or simply discard it. The technical data could be useful when solving problem situations such as the space available on the device or expected download times and thus costs.

With the exception of the technical details, the other information is easily present to the user on various stages of the listening process. On the selection list menus, the information is displayed next to the selected item, thus giving a preview based on which the decision could be made. This information is first shown about the particular subscription and later about the individual episodes. Lastly, the player view reveals additional details about the show while it is being played back.

By concentrating to the less technical information, emphasis was put on the human-to-human communication, even if we cannot guarantee the nature of this message exchange. After all, the author of the podcast shows himself not only in the content, but in the ways he describes the content within the podcast feed.

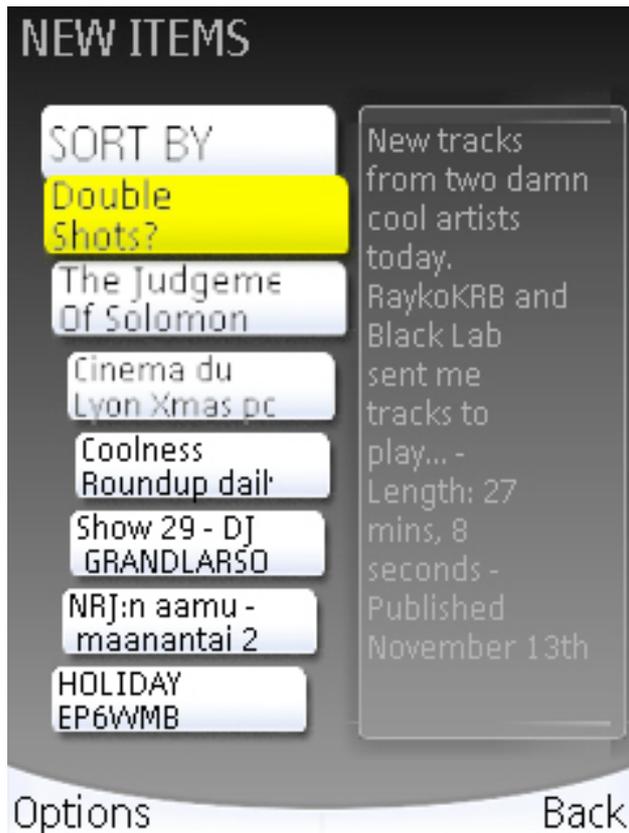
#### **4.2.5 Usability drivers**

#### **4.2.6 Basic usability of the prototype**

While the previous chapters describe the emotional factors, usability principles were taken into account as well. The mobile handsets are restricted by their size, both of the display and the input controls, and thus special attention needs to be paid to make controlling the application simple. Elaborate visual clues are hard to implement, as they require both screen estate and ways to navigate into them. The typical approach of using options menu to offer more functionality often leads to unorganized and lengthy lists.

One of the ways the prototype tries to help people to use it is to minimize the number of different keys that are needed to navigate inside the program and to perform functions. Also, the keys should behave the same way throughout the application.

The podcasts are organized into a hierarchy, which largely resembles the typical tree-structure of files and folders found on computers. This prototype offers some additions such as folders that create their content dynamically (such as the category “new items” that updates when a new episode arrives or an existing one changes its state. This is shown in figure 8. Despite the dynamic elements within the lists, the hierarchical structure is presented to the user.



*Figure 8. The prototype implemented groups that collect items from all of the feeds that match the set criteria. The “New items” screen collects all the podcast episodes that the user has not yet listened, across the podcasts that he has subscribed to.*

This tree is navigated by joystick. Up and down keys select previous and next items on the list while left and right keys move into and from the possible sub-groups. Right arrow and joystick presses do the same thing; they perform an action with the selected item. In most cases, the action is simply entering a sub-directory or playing a podcast. There are occasions where more than one option exists for any given element and in these cases a menu is brought up to let the user make a selection from the available options. The menu was designed to visually tie into the element, which brings it up; the selection remains colored in the background and the options are placed next to it. The most likely option for the element is selected by default and the alternative options are listed underneath.

The directory tree also mixes items and actions. Lists can be sorted by various metrics such as length, publishing date or their size and the sorting options

remains the topmost list item. These are separated from the rest of the elements by the size of the font (the actions are one-liners whereas the items typically use two). Lists do not loop, and thus reaching the sorting option can be done by continuously pressing the joystick up. Users do not need to pay attention to the time the key is pressed or pause at the suitable moment. The pay-off is that the last elements of the lists take more time to access, even if this can again be minimized by providing a fast scrolling of the list.

The player itself breaks some of the rules described earlier as a tradeoff to long-term usability and usage while moving. The left arrow key no longer leads back in the hierarchy, but instead the joystick is mapped to control the position and volume inside the given podcast. The option to return to previous menus is solely on the right soft key. Similarly the left soft key changes the displayed information in the information area of the player.

The added benefit is that the player can now be controlled even without looking at the screen. The keys can be felt through thin clothing and each press does not depend on the state the player is (excluding the left soft key that changes the information. This information is, however, of little use in the first place if you do not look at the screen). In this light, the resulting inconsistency was felt to be worth the possible initial confusion in the learning phase of the application.

Another deviation from the S60 style of navigation is the use of long key presses, e.g. key presses that last longer than a set interval (one second, in this case). These were used to fast forward and rewind the audio track while in the media player view. The functionality differs from short-presses, which skip to the next and previous podcast episodes. The result of such skipping is determined by the menu from which the user accessed the particular episode from. However, if the playing position is somewhere in the middle, the left joystick key first moves this position to the beginning of the current episode and another click is needed to reach the earlier one. The functionality in this regard is borrowed from Apple's iPod shuffle portable media players.

#### **4.2.7 Subscribing to podcasts and podcast removals**

A very integral part of podcast usage is to find the podcasts one is personally interested in and after that be able to subscribe to them. There are basically two methods that are most common; a directory based search that lets the user to browse a hierarchy (as of writing this, a rather familiar method was to use distributed OPML files) till he finds in interesting podcast. The alternate starting point is that the user browses the Internet from where he finds something that is interest to him.

The directory-based subscription can be seen as an extension to the directories that were available at the beginning of the Internet search engine. Yahoo, Lycos and others formed directories about web pages. These were later on replaced by active search engines, which crawl the Internet to create vast, dynamic databases that can be searched by keywords. This kind of searching has already been tried with audio content, but at the moment those services are not very common.

The application assumes the primary use case where the end-user browses the web and comes by something interesting that he then chooses to subscribe to. The feed is directed to the podcast application, which in turn adds the feed to the list of podcasts it should periodically and automatically check depending on certain variables. The identification happens via set feed handlers and mime-types. The application would set itself as a listener to a certain type and the web browser would then invoke it with the feed as a parameter.

The parameters used to decide when the downloads occur could include behavioral patterns as well as system status such as battery level, available space on the memory card and available connections to the outside world. Behavioral patterns might rely on the time of the subscription, time of the day and the observed update interval of the podcast feed. As a general rule, the podcasts should be updated when the device is on an available connection that generates the minimum cost to the user and is attached to a charger. They should be checked at the time when user has as few other activities as possible and be ready when he again needs his device. These cases form a believable scenario. At night time,

people typically sleep at their homes where the possibilities for available wireless network access and the charger connection are likely to be at their highest.

Battery level was designed to be the most dominating factor in choosing a suitable time for updates. The mobile terminal on which the application was decided to run is most importantly designed to be a phone. As such its primary function is to answer and make calls and if battery is low, less important tasks such as podcast downloads should be cut back unless specifically asked by the user.

Another already mentioned restriction is the available connection(s). Podcasts, which can be some tens of megabytes, are slow to be downloaded via GPRS connection and thus 3G or WLAN should be preferred. However, simply choosing the best available connection is not enough, as these may introduce costs to the user. WLAN connections are likely to be the safest to use, as the common billing models force the consent of the user. Either the connection is based ad-hoc (as an example on airport terminal while waiting a flight, one might go and purchase 30 minutes of time to the local WLAN) or the payment is done on other manners such as monthly fee. Cost based on the data rate is assumed to become more rare (this is, of course, speculation).

The latter part of the problem is how the podcasts are to be removed. This applies to the individual episodes as well as to the subscriptions.

The handling of subscription removals is less interesting, for that needs to be done explicitly by the user; after all, he was the one who asked for the subscription. The feed can be set to inactive stage, however, when the user has not consumed the provided content for certain amount of time or updates. Resuming from this state happens when the user asks for an update – this in turn leads into waiting period that is necessary while the podcast episode or episodes are downloaded. As such, if this sort of mechanism was to be used, it needs to be able to be set on or off from the preferences.

The episode removal is more the interesting matter. This is because of the size of the content, which can and is likely to be relatively large compared to the default available memory on the devices. As memory technology improves, this will be a

lesser problem but would eventually lead into usability problems, if not to anything else, such as the difficulty to manage and locate the interesting episodes from all the rest.

Also, to ask the user specifically to manage his subscriptions by deleting them manually is a both unrewarding and laborious experience. The nature of podcasts can be considered to be news and entertainment. Most of the time the content is not valuable after it has been listened through. Some content is or becomes “not valuable” to the listener even earlier than that. This does not apply to all content though, as an example, the episodes of “Learn French Podcast” can well be found interesting long after their first usage.

Thus I suggest and assume an approach that the podcasting client is allocated a memory space in which it can operate. This can be defined either in absolute terms (megabytes) or as relative value (60% of the memory card). If the space limit is not exceeded (which is to be tested with the header information of the download to see if the whole episode can fit and we do not need to do unnecessary download) the download is made automatically when decided by the aforementioned condition based system.

If there is not enough space for the new episode, a clean-up procedure takes place and it deletes the oldest, non-listened episodes till there is enough space for the new one if they are not marked as important (marked by the user as items that are to be saved). In the chance this will not be enough, the download is put on hold; the client shall not delete non-listened episodes.

It is to be noted that these mechanisms are not tested in practice nor are they implemented in the scope of this thesis. Separate studies would be needed to see how users deal with the abstractions described here; the users would not be bothered with downloads in progress nor with episode information gained from feeds unless the audio enclosure would not exist on the device as well. It is also to be noted that some parts of this behavior do need options and settings from the client itself, such as the possibility to save episodes from automatic deletion. These are taken into account in the reference design, even if they are not present.

## 5 User tests

### *5.1 User selection*

An external company recruited the test users and not all the desired criteria were met. The gender distribution was not as even as was hoped, six out of the eight participants were women. Also while each of the participants were asked to have some working knowledge about S60 environment, this proved to be on some occasions rather superficial and even the basic use of the platform was at times causing unnecessary problems in regards to the aims of this particular user test.

Each participant was also asked to have experience on browsing the Internet with a mobile S60 device. Whether this was WAP or WWW did not matter in this instance and mostly dealt with the requirements of the other tested projects. In the end, five participants stated they had used a mobile browser. The requirement came from external sources and dealt with other evaluations, which were held at the same time.

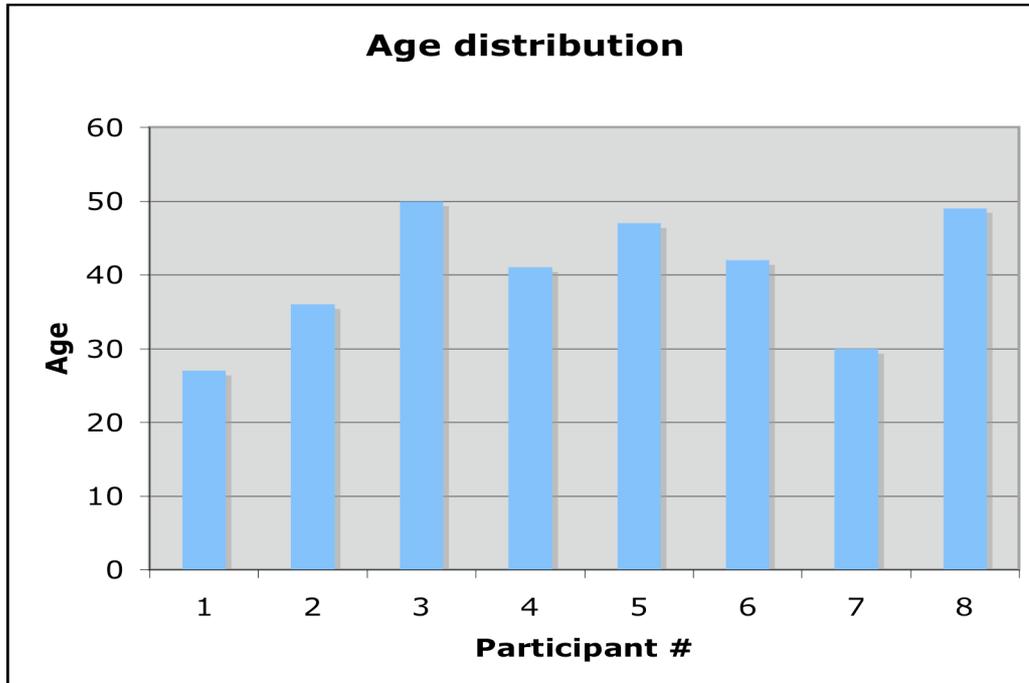
In regards of testing the podcast client, this was beneficial as podcasts are often distributed via the Internet. It is quite plausible that a common case to find a podcast feed is to stumble across the web page (discussion group or an advertising site) that gives the impulse for the user to subscribe.

The participants were not technologically oriented with a few exceptions. Two out of the eight had considerable technical background whereas the remaining six were not as fluent with digital devices.

The age distribution was rather wide, averaging about 40 years. The youngest of the participants was 27 years old while the oldest had turned 50. The exact ages of each individual user are found on figure 9. In general, the users were older than in the designated target group, in some cases considerably older. This makes it

harder to generalize the results for a younger audience. However, the participants were at least not more technologically adept than the targeted group of users.

Figure 9. The exact ages of the users were 27, 36, 50, 41, 47, 42, 30 and 49 years, respectively.



Every participant had at least a satisfactory level of understanding of read English language as they were asked. This was needed since not only were the content in English but the interfaces as well. The participants were not required to speak nor write English as they could comment in their native language (in this case in Finnish).

Six out of the eight participants were females whereas the remaining two were males. Only one of them had heard about the concept “podcasting” or “RSS” before. The occupations of the participants varied from cosmetologists to dentists.

## 5.2 Test process

The tests done were planned to be laboratory tests due to time and financial constraints. A two-hour time slot was reserved for each of the interviews and the events were lead by a usability specialist. In practice, the tests took less than two hours in total, to keep the fatigue levels of the users tolerable.

The compared applications ran on actual hardware. The Nokia Podcasting application was ran on a N80 multimedia computer as it did not run on the N95 prototype device despite best efforts. The prototype was made for N95 and relied on its hardware 3D acceleration features and thus did not run on N80. The devices were not greatly different and the test process did not involve tasks that would take the users outside the tested applications. The exact firmware version for the N95 was “week 44, 2006” whereas N80 used retail versions. The version of the firmware for N95 was important due to the prototype nature of the device and the changes that occurred in bi-weekly releases.

As the tests were limited by time, the podcasting content was pre-loaded into the devices. With the current connection speeds downloading the content would have taken a major portion of the reserved time for the interviews and would work against the philosophy of the delivery mechanism (e.g. the updates are supposed to happen in the background and mostly hidden from the user). Each of the compared applications and devices were prepared with the same content.

To help testing, the prototype did not implement the automatic updates of the podcasts. Thus, the application could be reset simply by exiting and re-starting it. This was not possible with the Nokia Podcasting application as its code base could not be tampered with.

A camera was installed on the test devices to transmit the image to the observer. Video stream was not recorded and the screen was visible to all persons in the room to avoid any confusion about what was being filmed. The image itself was focused on the display of the mobile device and did not show more than the fingers of the participants. Audio was not recorded during the interview either.

The tests were held in a private meeting room during weekdays. Two persons were present in the interviews in addition to the test user. One of the persons was a usability specialist who made the actual interviews while the other was part of the development team of the tested application. The interviewer was responsible of explaining the situation and led the users through the event.

The role of the observer was to take notes (with pen and paper) about the responses and reactions of the test person as well as to get first hand experiences how the users felt about the software. It was not revealed that the observer was the developer to make sure the users would not be too kind in their replies. No audio-visual recordings were taken during the interviews, again to make the situation more comfortable to the test users.

The interviews were given tasks to perform on both of the applications and the order was altered so that half the test users started with Nokia Podcasting Application and the remaining half with the prototype. To further improve randomness, the first two used first the prototype, following four the Nokia Podcasting Application and the last two started again with our prototype. The two more technology-oriented test users were made to start with different applications.

Some of the planned test tasks were removed from the test process to make the situation more pleasant for the test users and avoid embarrassing them. Tests were held in laboratory, which caused some social restraints and some possible tasks such as walking around the room might have felt too awkward. Thus only tasks that the user could accomplish while sitting were given to test participants.

Eventually the users were asked to perform five tasks:

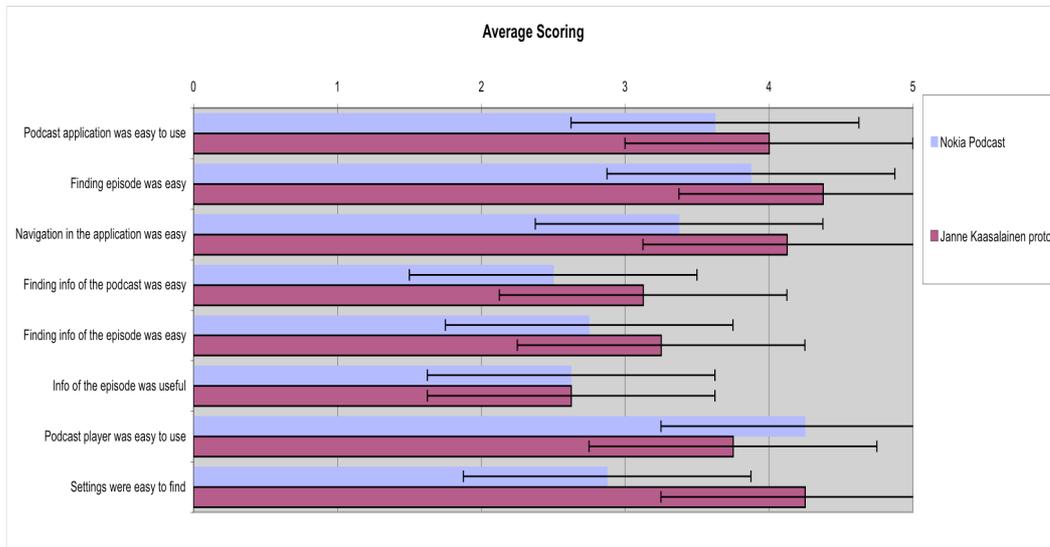
1. Listen the latest episode from “NRJ” podcast.
2. Stop the listening and return to podcast menu.
3. Listen the third newest episode from the podcast “Whak my Bush”.
4. What information can you find from this episode?
5. How many new episodes do you have in your mobile that are ready to be listened?

After following through the task assignments with both applications, the users were asked to fill in a questionnaire and evaluate their experiences. Also, an informal discussion was held about the experience.

### ***5.3 Test results***

The users were asked to evaluate various aspects about each of the applications by giving a rating from 0 to 5. 0 was reserved to indicate the most negative answer possible (e.g. “settings were impossibly hard to find”) whereas 5 meant an extremely positive experience (e.g. “settings were in obvious place”). The averages of the scores as well as the test questions are indicated in the figure 10. Podcast application specific questions missing from figure 10 were “Could you imagine yourself using podcasts in the future with this particular application?” and “What kind of a feature you would have liked to have?”

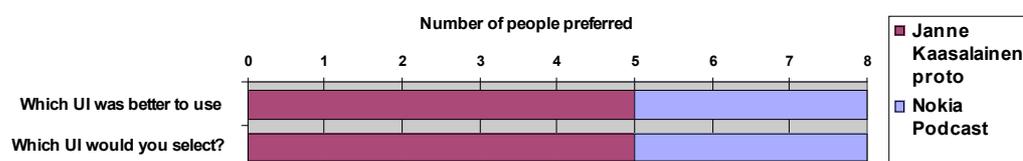
Figure 10. The bars represent the averaged scores for each of the questions the users were asked to evaluate. Blue bars (light grey) indicate the scores given to Nokia Podcasting application whereas the red bars (dark grey) present the scores the tested prototype received. The variation in replies is shown with black lines.



The Nokia Podcasting application is consistently with one exception slightly behind the prototype implementation. The margins are, however, relatively small. The prototype lost only on the evaluation of the easiness of the music player, tied in regards of the useful information presented to the user and won the remaining tests (six tests out of eight).

The last part of the questionnaire dealt with two questions asking the users to make a choice between the applications. The values in figure 11 present the number of users that made the particular choice in regards to each of the questions.

Figure 11, the preference choice of the participating users. Five users out of eight preferred the prototype. The remaining three would have chosen the Nokia Podcasting Application.



For the question “which UI is better to use” three of the eight participants answered the Nokia Podcasting Application whereas the remaining five chose the developed prototype. The numbers for the question “which UI would you choose yourself” were the same.

In total, the users had to answer 27 questions, ten about each of the compared applications and seven that compared the two and tried to gain understanding about their preferences. The remaining questions that have not been mentioned so far were:

1. How important to you is the pleasure of using the interface (on scale from 0 to 5)?
2. How important for you are the features (on scale from 0 to 5)?
3. Which were the three best things about these applications?
4. Which were the worst things about the applications?
5. What would you wish to change or improve about these applications?

In general, the answers highlighted the role of the user interface over features, but such result is likely to be very culturally and demographically depending. That said, pleasurable UI averaged to 4.5 whereas the importance of features got an average of 3.4.

#### ***5.4 Findings and the analysis of the results***

The user tests were organized together with two other projects and thus the selection of participants was a compromise. Also, the tests were 1-2 hours in length, which forced the study to concentrate on the immediate usability, and experience what the UI was able to offer. Podcasts and RSS, by their nature, are designed to help people manage information flows over longer-term durations and to filter quickly the irrelevant and relevant articles. Thus the likelihood of getting irrelevant feedback on the usability was to be taken into account. Short-term

usability tests where users need to move from a fixed, familiar interface style to something they find completely new are more likely to cause resistance and annoyance when things do not work as they did before and after the specific tests.

This also goes the other way around to hide possible problems the prototype UI might have over a prolonged usage. Factors such as the visual pleasantness and “wow”-factor can hide usability issues as well.

The other projects wished for experienced S60 test users, and thus the S60 habits were likely to be strong in the user group. Also, most of the users were older than the approximated target group, which is a factor that should also be taken into account while analyzing the results.

In interpreting the test results it should also be kept in mind that the prototype was an initial one and was missing even many planned features such as indicators for the length of an individual podcast. Details such as volume control had not been fully ironed and those were inevitably affecting the user experience in their feedbacks (controlling volume is, after all, rather crucial element of playing audio).

Despite the request that the users would have been experienced users, the settings were difficult to find with the Nokia Podcasting Application. This was partly due to the fact that the settings only emerged into the options menu at the root level of the UI structure and that the options menu itself was relatively long one. Nonetheless this is the default S60 method to place the settings. The prototype, on the other hand, did display settings as a main level category, so the easiness to find it was understandable. This is, however, important finding as it shows that the path to access the settings need not to be available everywhere in the software and the inconvenience such as traveling back through the podcast hierarchy is understandable and logical.

The animations used to change from one view to another were perceived fast. The alternative application did not use animations and view changes occurred with slight loading delay. The user perception was more positive when there was visual feedback of the software reacting to the user. It seems that when the user makes

larger context switches, the short animations are justifiable if they can explain what is happening (even the mere presence may indicate a context switch to the user).

One partly expected finding was also the separation of user opinions if volume adjustments should work on left-to-right axis or on up-and-down. Some justified their preferred direction with their expectations how a remote control for their TVs work (which do differ between brands and models as well). It is to be noted that the selected users were hoped to have previous S60 experience which defines the volume to be adjusted horizontally. This is likely to affect their answers, even if only subconsciously. What this does show is how hard it is to change user habits and how deep the expectations from previous systems affect the experience (after all, the chosen direction is an arbitrary decision). Due to consistency within the platform, the S60 should be used as a reference in future prototypes when designing applications for shipping products.

The player part of the prototype was in fact the most problematic for the test users. The volume problem was obviously one of the main culprits for this, but in general much of the functionality was simply not implemented in time. Indicators for volume and textual metrics for time (current play position, total length of the podcast) were not available to the users due to the time constraints with the implementation. Also, the player did take advantage of the four-way joystick in ways that are not particularly usual for S60 applications. The application used the joystick itself to control the software instead of moving the focus to the element that was to be controlled.

The emotional contact between the test users and the content was understandably thin, as the users did not have practical means to have content they would have liked. The last two questions, however, were crafted to make the users give an evaluation (without justifications) of which of the applications they would have chosen themselves. This was deemed to be the most important finding on the test, and on that the prototype was the users preferred choice even if with a small margin. The technical immaturity was likely a factor that the margin was not larger than the results demonstrate.

As some of the users were highly non-technical, the comments and feedback were not always in relation to the software itself but the platform it run on. This is to be expected, to some extent, but it also highlights that the user experience does not rely on the software or hardware but the combination of both. In optimal case, neither should be omitted when designing the products and both should support each other given the targeted users.

The evaluation of the user experience was eventually shrunk to the two last questions (“Which UI was better to use” and “Which UI would you select”). Of these the latter was deemed more important as that would give the most meaningful answer about the preference of the users. The study did not answer to questions how the different elements affected to the user experience or which of these elements were most meaningful ones.

Overall, the prototype scored well against the established application. The scores on their own did not differ much between the compared clients, but they were consistently in favor of the prototype. Users preferred the prototype solution in eight of the ten questions. Qualitative comments also supported the general preference drawn from the measured scores. The 27-year-old female podcast users said that the current (Nokia Podcasting Application) UI is ugly and boring, and thus would not want to use it. On the other hand, the 53-year-old female told the interviewer that even if she prefers the current UI the prototype could be better in the long run.

### ***5.5 Improvement ideas***

The most immediate interface improvements should be done to the player part. Most notable of these are the inclusion of volume control status and the numerical information about the length of the playing podcast and the play position at a given time. These were mostly known issues, however, and established already in the designed screenshots.

Technology-wise the next step would be to actually implement the background mechanisms that deal with the podcast downloads and removals so that the demo could be made usable on longer-term tests. The importance of such studies should

be highlighted as in these short laboratory tests the emphasis was on learnability. The aspects of the look'n'feel were also emphasized. Based on the results it is hard to generalize the user feedback in regards of long-term usage. Podcasting tries to help the user to stay up to date with new episodes instead of manually downloading the episodes when the need occurs.

The volume control should be re-thought, even if not necessarily changed in the future. As stated previously, the current implementation knowingly broke the user interface style of the current S60 devices. The justifications for this have been mentioned elsewhere. What should be studied is what the user think of the method after all the other functionality is included to the controls. As seen in the study about other similar products, the direction of the volume control varies and is not always consistent even inside a manufacturers own line-up (Apple iTunes and iPod Shuffle and iPod Nano for an example). Many AV systems, such as TVs, do use the vertical dimension for volume control.

As the prototype was the preferred choice for the participants, further studies should be done to break down the user experience factors by altering the elements present in the interface. Such studies would hopefully clarify the relative importance of the functions and the looks that the prototype implemented.

## 6 Conclusions

As a result from this thesis a prototype implementation of a podcast client user interface was created and validated in user trials against a competing product. The implementation offered the basic functionality that was deemed important to study user experiences and interface engineering over a short period of time and with non-technical users of which most had never seen a podcast application before.

Moreover, the thesis studied the possibilities to offer additional value to the end user in subtle forms and prioritize the default information presented in the user interface. Most of the methods were not functional features per se, but were implemented to create a more pleasing experience.

The user studies were made with a limited set of users to compare the prototype to an official podcasting client and made to evaluate the pleasantness of each of the applications. The majority of the test users (five out of eight) found the prototype to be their preferred option if it was a fully working one. Main reasons for this seemed to be fast and responsive interface that presented relevant information in easily understandable manner. This was visible, for example, in the scores for easiness of navigation. The look of the application was very likely to affect the positive feedback as well.

It could be speculated that a holistic user experience is a balance of at least usability, pleasure of use, aesthetics, responsiveness and predictability. Other factors are likely to come into play as well, but none of the aforementioned aspects suffices alone. Some level of usability is a requirement, but does not guarantee that the product is liked. Equally, an appealing visual look does not help long if the user does not understand the system. Thus, the design needs to take each aspect into account and find a solution that takes these into serious consideration.

It is extremely difficult to say what aspects affected the positive end results when it comes to user experience. But what is clear is that it is not enough to think about application design with mere functional demands and requirements; how things appear and feel is an important factor to the actual end user.

It is difficult to name a formal method of creating attractive and pleasant applications from the end-user perspective. Also, the proposed hypotheses about emotional functions (transferring emotional context and creating a connection via quirks) are not validated by these tests alone but they are not shown to be in direct conflict either. Further studies would be needed to refine the assumptions and to see how general those might be.

With its own small part the prototype has paved ways of what is to come. Some of the principles are already implemented in new S60 phones, even if the credit may be due to quite different parties than the author of this paper. For example, Nokia N95 includes a new media player that also implements the controls that are directly accessible with joystick directions. In fact, the controls are the very same as proposed in this prototype implementation with the difference that the wheel is rotated 90 degrees. Album art is also becoming more widely used in the particular media player.

## References

Adesemowo , A. Kayode; Tucker, William D. (2005) Instant messaging on handhelds: an affective gesture approach. South African Institute for Computer Scientists and Information Technologists, July 2005, 8 pages

Alben, L (1996) Quality of Experience: Defining the Criteria for Effective Interaction Design. Interactions, Volume 3(3), May/June 1996, pages 11-15

Anderson, Chris (2006) The Long Tail: Why the Future of Business Is Selling Less of More. Hyperion, July 2006, 256 pages

Apples additions to RSS 2.0 specification, <http://www.apple.com/itunes/store/podcaststechspecs.html>, reference at 29.3. 2007

Butler, Keith A. (1996) Usability Engineering Turns 10. ACM Press, January 1996, 17 pages

Egri, Lajos (1972) Art of Dramatic Writing: Its Basis in the Creative Interpretation of Human Motives. Touchstone, February 15, 1972, 320 pages

Grady, Helen (2000) Approaches to prototyping: Web site design: a case study in usability testing using paper prototypes, IEEE Educational Activities Department, September 2000, 7 pages

Halvey, Martin; Keane, Mark T.; Smyth, Barry (2006) Mobile web surfing is the same as web surfing. ACM Press, March 2006, 6 pages

Hassenzahl, M; Tractinsky, N (2006) User Experience – Research Agenda. Behaviour and Information Technology, Volume 25, No 2, March-April 2006, pages 91-97

Kuan-Ta Chen, Chin-Laung Lei (2006) Understanding player behaviour for game design: Network game desing: hints and implications of player interaction. ACM Press, October 2006, 9 pages

Lim, Youn-kyun; Pangam, Apurva; Periyasami, Subashini; Anjea, Shweta (2006) Comparative analysis of high- and low-fidelity prototypes for more valid usability evaluations of mobile devices. ACM Press, October 2006, 2 pages.

Mehrabian, Albert (1981) Silent messages: implicit communication of emotions and attitudes, Albert Mehrabian, 2. Ed edn. Wadsworth, Belmont, Calif. 1981, ?? pages

Milne, Mike (1998) Beware of the QWERTY. ACM SIGGRAPH Computer Graphics, Volume 32 Issue 1. ACM Press, February 1998, pages 8-10

Mizutani, Michihito (2006) Emotional Communication. Media Lab, University of Art and Design Helsinki, March 2006, 51 pages.

Mäkelä, A.; Fulton Suri, J. (2001), Supporting Users' Creativity: Design to Induce Pleasurable Experiences. Proceedings of the International conference on Affective Human Factors Design, ASEAN Academic Press, 2001, pages 387 - 394

Nicolas Ducheneaut, Nicholas Yee, Eric Nickell, Robert J. Moore (2006) Games and performances: "Alone together?": exploring the social dynamics of the massively multiplayer online games. ACM Press, April 2006, 10 pages

Nielsen, Jakob (2000) Designing Web Usability, New Riders, cop 2000, 419 pages.

Podcast, <http://en.wikipedia.org/wiki/Podcasting>, referenced April 1<sup>st</sup>, 2007

Reponen Erika et al (2006) Mobile Video Recording in Context, interactions, July & August. 3 pages.

Roto, Virpi (2006) Web Browsing on Mobile Phones – Charasteristics of User Experience, Helsinki University of Technology, December 8<sup>th</sup>, 2006, 86 pages.

RSS 2.0 documentation, <http://blogs.law.harvard.edu/tech/rss>, referenced at 29.3. 2007

RSS 2.0 example channel, [http://en.wikipedia.org/wiki/RSS\\_\(file\\_format\)](http://en.wikipedia.org/wiki/RSS_(file_format)), referenced at 29.3. 2007

Rudd, Jim; Stern, Ken; Isensee, Scott (1996) Low vs. high-fidelity prototyping debate, ACP Press. Interactions, Volume 3 Issue 1, 10 pages

Sánchez , J. Alfredo; Kirschning, Ingrid; Palacio, Juan Carlos; Ostróvskaya, Yulia (2005) Towards mood-oriented interfaces for synchronous interaction. ACM Presss, October 2005, 7 pages

Trewin, Shari (2006) Physical Usability and the Mobile Web, ACM Press, May 2006, 4 pages.

UPA (Usability Professionals' Association): "Usability Body of Knowledge", <http://www.usabilitybok.org/glossary>, referenced at April 1<sup>st</sup> 2007