Web Portal for a Wireless Infotainment Platform

A concept implementation of a portal managing media content later accessed from an embedded unit in a vehicle.

Master of Science Thesis

NILS HEUMAN

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, March March 20099

The Author grants to Chalmers University of Technology and University of Gothenburg the non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet.
The Author warrants that he/she is the author to the Work, and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Web Portal for a Wireless Infotainment Platform
A concept implementation of a portal managing media content later accessed from an embedded unit in a vehicle.

NILS HEUMAN

© NILS HEUMAN, March March 2009.

Examiner: STAFFAN BJÖRK

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden March March 20099

Abstract

Mobile internet is becoming popular and appears in many devices, but has not yet penetrated into the automotive industry. One business idea is to place a computer in a vehicle. The computer equipped with interfaces for connection to internet, external storage units and an easy to use graphical interface. This report describes a project developing a concept where a system is built that distributes media from third party content providers on the internet to the embedded unit. The specific task is to solve how it should be possible to do configuration with a few clicks on a desktop computer and then have media available in the vehicle. The report explores what services can be integrated in such system and the outcome is a proof of concept prototype that shows the functionality.

Contents

1. Introduction

Today we have media all around us, the communication services are getting more complete and you can reach a fast internet connection almost everywhere. This opens a lot of new unexplored opportunities. In this report services for acquiring media such as music, movies and news in a car is reviewed.

Cybercom Group [10] has been part of a research in this area where the integration of mobile phones, MP3-players etc in cars has been studied. A platform has thereafter been developed and basic software functionality for the platform has been created. This platform is called WIP (Wireless Infotainment Platform). Cybercom Group now wants to create a concept that explores what more possibilities the platform can offer. The concept is a layer in form of a web portal between the WIP and content providers on internet that should be managed from the end customer's desktop computer.

Most vehicles today only offer radio, media on CD and MP3 for entertainment. The music systems in cars are often improved in the aftermarket by adding screens and computers for showing video; this could be done already in the production and with integrated services for retrieving new content without having to move a portable unit.

1.1 Purpose

The purpose of the project is to see what possibilities there is for managing media online that later should be viewed in a car. No such service exists today but companies have started to move in that direction. Examples of this can be seen in BMW-cars where a tiny screen shows and let you choose between CD, radio and TV by a simple interface in the center console of the car [23]. Another example of a service that has a large market share today is iTunes where you administrate your MP3-player on your PC and have the possibility to plug it into your car [28]. The car then integrates the MP3-player into its media system so track titles can be viewed in the CD-player and track selection can be done from the steering wheels media buttons. This interface is for example available in new Volvo cars [2]. A prototype that combines these two technologies need to be built to explore what market segments that exists and makes the effort for the customer smaller to get a complete media experience.

What media services are suitable to access from an embedded unit in a car?

To answer this question a study and model over what services that exists and how they can be implemented could be done, but Cybercom Group wants a working system that they can test run and show for potential investors. Also if not implementing a system it would be hard to get all details of what problems that could occur. No system with this functionality exists on the market today and for that reason deeper studies of other systems cannot be done.

1.2 Delimitations

There have been projects that have tried to implement and integrate many types of functionality on a car computer platform; this has however failed. A lot of investments have been done in those telematic projects where position and integration with the car was supposed to be created. After their failures there have been a stop in such projects and more limited approaches in functionality has to be taken [15]. Therefore has this project been limited to only handle music, web radio, podcasts and RSS-feeds. Depending on time available other services as applications, widgets and more advanced services as charging for parking tickets, booking of service of the vehicle may also be investigated and implemented. The design should be done to be prepared for expansion with all the mentioned functionalities. The prototype server system will not reach the end customer so a minimum amount of time should be spent on graphics and performance, still the system should be esthetic enough to attract investors. The system will be demonstrated internally so security has a low priority.

One technical question that was discussed in the startup of the project was how to handle a client that loses its connection to the internet and the portal. The decision was made that the client always is online but that the system should be designed with offline mode in mind. The prototype does not need to implement offline mode.

1.3 Targeted Readers

The target with the report is to explore which services that is reasonable to include in a car media distribution platform. This input can later be used in a bigger project that can take care of all these. After having read the report and tested the application a large part of the investigation work of such service should be settled. The target industry is the personal car industry and the project is a proof of concept.

1.4 Report Overview

The report is divided into seven chapters, first an introduction about why the project has been started. More detailed background information is then given and a brief background of the company and project. To clarify some of the services that exist in the marked they are described. Technologies that are used in the project will be introduced shortly to give a better understanding when reading the report.

The third chapter shows the planning and describes the method that is supposed to be followed to maximize the result from the project. It also describes which people that are involved in the process and their responsibilities. A short overview of the different parts of the process is also shown.

The development chapter describes what was done in the project. Each iteration is gone through with detailed information and describes problems that occurred.

The fifth chapter shows the result, mostly the web portal but also a short part that describes how far the client development has gone when this report is written.

In the discussion chapter the process and result is analyzed to see what that could have been done different to have a better result. Questions and ideas that appeared in the project are also discussed.

The report ends with the conclusions drawn throughout the project and guidelines on how future projects could be constructed.

2. Background

2.1 Brief history of Cybercom Group

The Cybercom Group is a consultancy company that offers end-to-end solutions. It is a supplier in security, portal solutions, mobile services and embedded systems. Cybercom also offers strategic and technological expertise in the following markets: industry, telecom, banking, media and financial services, retail and the public sector. [10]

Cybercom was founded in March 1995 and have since then been focused on growth. The growth has been both organic but also through strategic acquisitions. In 2007 Cybercom acquired auSystems from the Teleca group which resulted in sales and head count more than doubled. Through this an office in Gothenburg was acquired. [10]

Cybercom employs about 2000 persons worldwide in 26 offices in 10 countries. Four of them are located in Sweden and one in Gothenburg. The office in Gothenburg is divided into three main divisions: Automotive, Embedded and Enterprise with a total of 200 employees. [10] This thesis is made in the Enterprise division but the project is a mixture of people from Automotive and Enterprise.

2.3 Project background

Figure 1 explains the cornerstones in the concept. This thesis regards the *Car services Portal* shown in the figure. The portal should allow a customer use its desktop computer browsing to *My Car Homepage* located at the server on the internet, and through a graphical interface select content from third party providers. This content is then transferred directly to the *Cybercom WIP Browser* and is present when the customer enters the car. The *Car manufacture system* represents car salespersons that should have access to a backend of the portal with the possibility to pre configure the selections of content in a newly sold car.

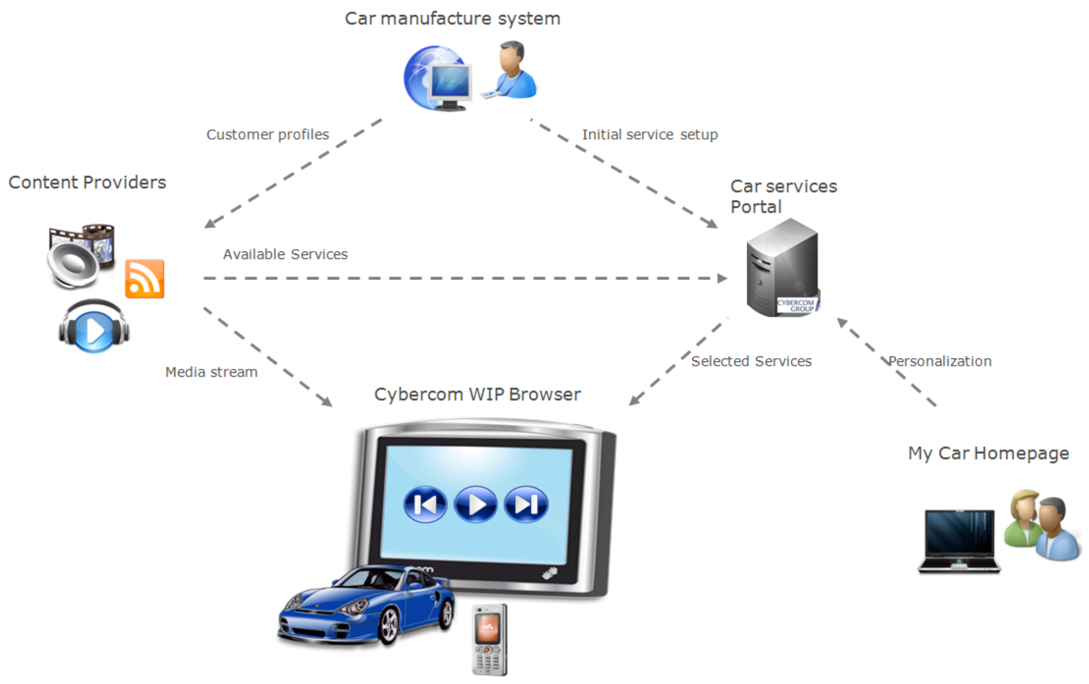


Figure 1: The concept describing how the whole system with all parts would look like.

2.4 The current WIP-platform

The existing hardware platform that has been developed has support for playing sound and video, connect to internet through different interfaces, handle connections to external devices as mobile phones and external hard disks. The specifications for the platform have been developed after a detailed study regarding an Open Platform for Nomadic Devices [27]. The study has researched the possibilities there are for an embedded unit in a car and what demands customers have.

The existing software that runs on the platform can play music, retrieve and show news. The interface is quite simple and not very esthetic. The hardware platform will be changed to handle a higher resolution and more colors and this project will use the new platform.

The platform is natively running the framework QT [19] for creation of user interfaces and a Java environment can be loaded for running Java programs.

2.5 Services

To get a better view into which services exists on the market which offers similar functionality as the concept a few will be shortly described in the following sections.

2.5.1 iTunes

iTunes is mainly a service for administrating media on a portable device as an MP3-player called iPod. By time the service has expanded and now there is support for more advanced media devices that handles video and applications. iTunes Store offers the possibility to buy music on the web, subscribe to podcasts and buy audio books. The service requires that the user downloads an application that connects to the web for downloads and payments. The application is also needed for transferring the media to the MP3-player. The company Apple stands behind iTunes. [3] The sales of iPhone 3G, Apples latest device connected to iTunes, has increased compared to their non 3G phone. Almost seven million devices have been sold in one quarter and that is more than the sales of the old phone for the last five quarters combined. The overall sales for the company have also increased. [25] The iTunes app store where applications can be bought and downloaded for free was released in July 2008, in September the same year there were more than 3000 applications available and users have downloaded more than 100 million applications. [26]

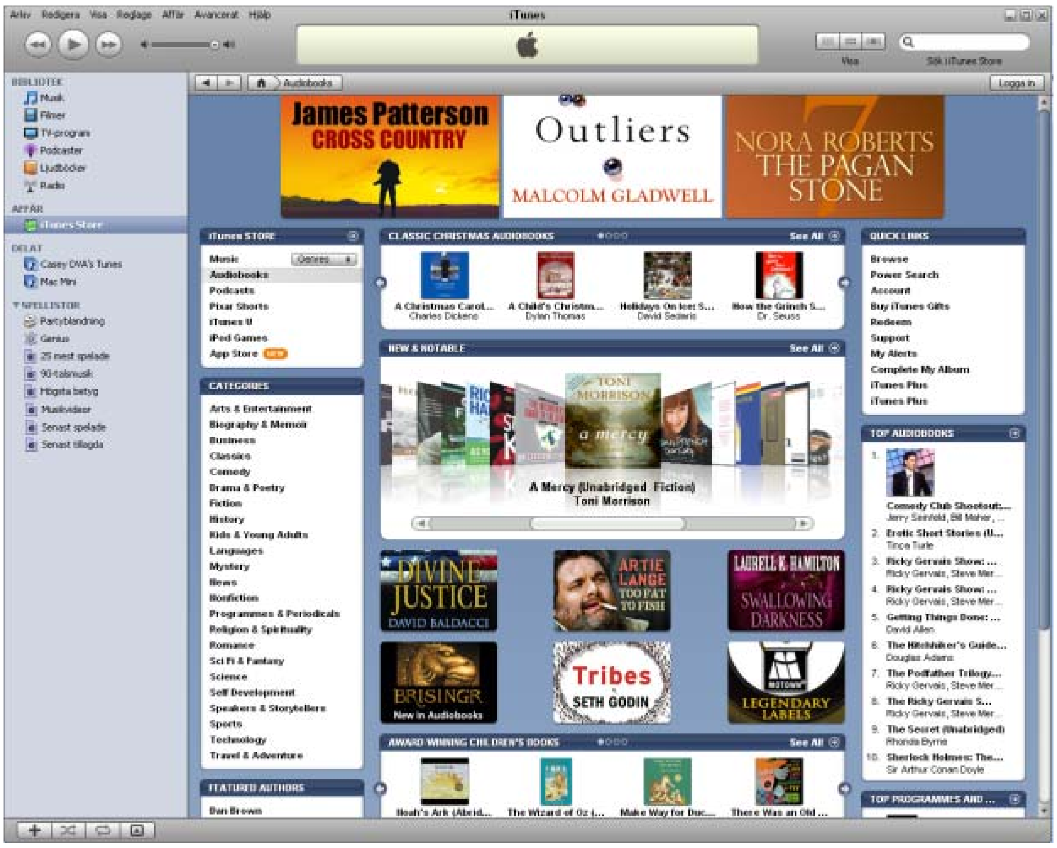


Figure 2: iTunes application.

2.5.2 Sony Ericsson Play Now

Play Now is a service that is reached from Sony Ericsson mobile phones. It allows user to easily buy music, ringtones and applications. It also offers the ability to send a short recorded section of a song for identification its title. [4]

2.5.3 MP3.com

Mp3.com distributes free music and media. Artists can create accounts and spread their own music through the site. The site also provides podcasts and videos. Some of the music on the site is however not for download, only for preview. [7]

2.6 Technologies

To give a better understanding of the technical parts of the report a few technologies are described in the following section.

2.6.1 RSS

RSS, (Really Simple Syndication) is an XML format that often is used for distributing news in so called RSS- and Podcast-feeds. It specifies some standard tags that has to be included and some that are optional [8]. Figure 3 shows an example of how news RSS feed is built. First a description of the document is set with the default tags, then for each element that the feed links to (in this example the <link> is pointing to a page showing the news text).

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <rss version="2.0">
  <channel>
      <title>GP/Göteborg</title>
    <link>http://www.gp.se/</link>
    <description>Västsveriges största morgontidning!</description>
    <language>sv-se</language>
    <image>
      <title>Göteborgs-Posten</title>
      <url>http://www.gp.se/gp/road/Classic/shared/images/goteborgsposten_144.gif</url>
      <link>http://www.gp.se/</link>
      <width>144</width>
      <height>19</height>
    </image>
    <managingEditor>bosse.dahl@gp.se</managingEditor>
    <webMaster>webmaster@gp.se</webMaster>

<item>
  <title>Julstaden har invigts</title>
  <link>http://www.gp.se/gp/jsp/Crosslink.jsp?d=113&amp;a=462745&amp;ref=rss</link>
  <guid>http://www.gp.se/gp/jsp/Crosslink.jsp?d=113&amp;a=462745&amp;ref=rss</guid>
  <description>Julstaden Göteborg invigdes klockan 17 på fredagen av Göran Johansson.
  </description>
</item>
…
```

Figure 3: Example of how news RSS feed could look like. [14]

2.6.2 QT Cross-Platform Application Framework

QT is a cross platform application framework that the client in this project will use for its user interface. The QT webkit has been integrated and allows to use a web browser and to create modules for it in c-code. The browser Arora [20] is built with the QT library and can be used as a reference test browser. Interfaces built with the QT library are based on an XML format and the interfaces can be styled with certain style files. The style files have similarities with CSS (Cascading Style Sheets). [19]

2.6.3 PHP

PHP (PHP: Hypertext Preprocessor) is a scripting language used on the web, much of the syntax is borrowed from C, Java and Perl. The language does not demand variables to be typed or declared. PHP is freeware and runs on both Windows and UNIX platforms. It uses an interspreter that is called from the web server when files with the extension php are accessed. [21]

3. Planning

This chapter will describe how the project is structured, which methods is planned to be used and how the work was planned to be carried out.

3.1 Time Plan

The time plan of the thesis is shown in figure 4. In the project startup the concept was to be translated into requirements, and an introduction into the company systems was planned to be done (A in the figure). Then three months of full time work on the development was planned to follow (B). After this development phase the project and thesis work was planned to continue besides full time studies for two months (C) and finally two months besides half time studies where the report should be finished and presented (D). The goal was to have a running application for demonstration at the end of the three months of full time work. Afterwards minor adjustments was planned to be done as finalization and documentation. The time plan when defining the project was based on four weeks for introduction, twelve weeks of development and four weeks for report writing and preparation of presentation.

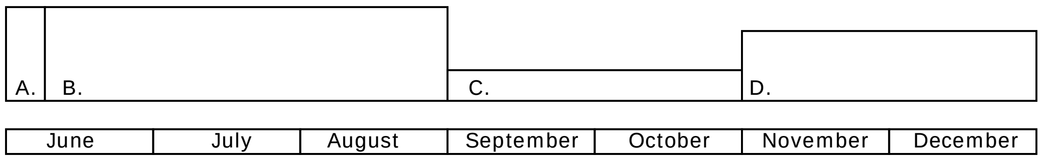


Figure 4: Showing the time plan, height of boxes represent work intensity.

3.2 Overall methods

The development of the new platform was planned to be done by two persons, one responsible for the server platform (which this thesis is based on) and one responsible for the client. The work was planned to run concurrently, specifications for the communication between the server and the client was planned to be decided by the developers. The specifications for the systems overall functionality and purpose was planned to be decided in meetings where a steering group is gathered. The group was planned to include salesmen and developers, both the ones that was supposed to develop the new system but also developers that have knowledge of earlier projects on the platform. A base idea existed of what functionality the platform was required to handle. Depending on limited amount of time for this first implementation of the portal, functionality requirements was planned to be added or removed to keep the project within the time plan. The aim was for as much functionality as possible but the most important criteria is that the service would be up and running for demonstration at the end of the project.

Many different software processes exists; this project can be put under the category rapid software development process [5] since much functionality should be implemented in a short time. The server part was not planned to be put on a production server and the specification of the project was not planned to be detailed or defined when the development was planned to be started. A waterfall model where all requirements defined at project start is therefore not usable; instead agile methods where requirements are specified at the same time as the system is implemented are more suitable. An iterative process where the first iteration is a prototype that will be thrown away was planned to be used [5]. The further iterations were planned to build on each other but with new requirements.

The steering group takes decisions of which functionality should be implemented was supposed to regularly inspect the progress and have new or changed requirements. The project was planned to be internal and no real release date for delivery was planned to be set, neither there was plan for a solid specification of what the project should contain or how it should be done. Therefore an evolutionary model [5] was planned to be used where the steering group can inspect the progress and propose changes as they see the system evolve.

The aim for the development phase was planned to have running systems which could communicate so neither the client or the server side application development would be stalled because of lack of functionality on the other side. The progress can be divided into a few phases, the Pre Study where research was planned to be done and the overall systems specifications are set. A concept phase which end up showing a simple system that shows some of the functionality searched for. The concept phase was planned to get hands on example to understand more about what is easy and quick to implement but also to get a basic view of how the systems user interface can be designed. This was planned to be made to give the steering group a better view for extending the specifications. This is a method that is recommended by for example MacCormack [1].

3.3 Pre Study

The beginning of the project was planned to include a pre study where services that offer similar or coveted functionality are analyzed to see if they could be merged or in some way integrated into the system. Services that the platform might contain should be analyzed to see how hard it would be to implement them in the system. Platforms for the development was planned to be chosen as well as language and tools, both for the client and server. The communication between the client and server was planned to be defined. The older systems was planned to be studied to see if any functionality or code could be reused. Conclusions based on earlier reports was planned to be acquired for minimizing the hedgehog syndrome [6] where errors from former projects are repeated due to lack of knowledge about them.

3.4 Concept Phase

In the project startup meetings that regard the functionality was planned. From these meetings vague requirements was planned to be defined which specifies a simple prototype system. The prototype was planned to be a simple web portal that administrates the media which is planned to be implemented and to get a clearer view of how an interface that maximizes the user experience should look and work. The focus was not planned to be on the design or performance, the idea is to find flaws in the design and get a better understanding of the technologies and difficulties involved in content management. Based on the concept more detailed specifications of the system was planned to be defined by the steering group.

3.5 System Phase

The design and implementation of the actual system was planned to be started when a new specification had been created. The development was planned to have a few natural iterations since the other project members was not planned to be available all the time of the process. When they are available again the system was planned to be reviewed and more specifications was planned to be set. As the system evolves regularly reviews was planned to get the status of the development and to add new parts to a priority list.

4. Development

This part describes how the project was executed and how the development went. How specifications were decided and what they include.

The project started with a concept about how a whole system that distributes media to a car and background study of this area. The goal was to create a system that could manage media that were distributed by third part to an embedded media platform in a car. The information should be transferred over the internet and the media should be selected by the customer from its desktop computer.

4.1 Initial Planning

An initial meeting was held with the steering group where some basic specifications and deadlines were set. The following requirements were set on the server application:
A web portal which is accessed from a desktop computer where the user can select the following media:

- Sony Ericsson media
- iTunes
- Web radio
- Audio books
- News – RSS
- NewsML[1].

The Web portal should have a fresh and cool Graphical User Interface; there should be settings for changing themes. Car salesmen should be able to access standard configurations to setup user accounts at sale. The portal should be compatible with Windows CE[2]. There should be support for downloading applications.

A lightweight portal which is accessed from the car should also be developed; the lightweight portal is from now on called the WIP Portal and should give the user some ways of adding or removing content. In the initial steering meeting it was stated that progress should be visible in the forthcoming week and that a preliminary result should be done in the nearest four weeks.

A demonstration version should be available three months later and then changes will be done forthcoming.

The reason for having the portal to administrate the content is to simplify the interface on the client and to minimize the administration done on the road. Ideas were proposed to lock the interface when driving; this could be implemented in the client.

[1] XML standard for distributing news [11].
[2] Windows CE is an operating system from Microsoft mostly used on embedded units [22].

The graphical user interface should have a menu tree of the content; the links should take the user to a listing where content can be selected. The user should also be able to overview the selected content.

Technologies were decided in the meeting. The web portal will be built on PHP and either MySQL or MSSQL. After the meeting system environments were setup, first IIS (Internet Information Server) was used as web server since it was company standard. Problems with the server were however detected quite soon so instead of struggling to get it to work again and based on earlier stability issues with IIS the web server Apache was installed instead. MySQL was selected as database engine, to make the system build on a genuine open source platform. The development platform was a laptop and the development was done locally. The web server placed on the laptop was still accessible over the internal network so the client developer could access it for testing. When selecting the database engine a guideline in the development was set to avoid using MySQL specific queries and keep the opportunity to run on other systems open. Therefore SQL queries have been separated from the PHP functions as much as possible.

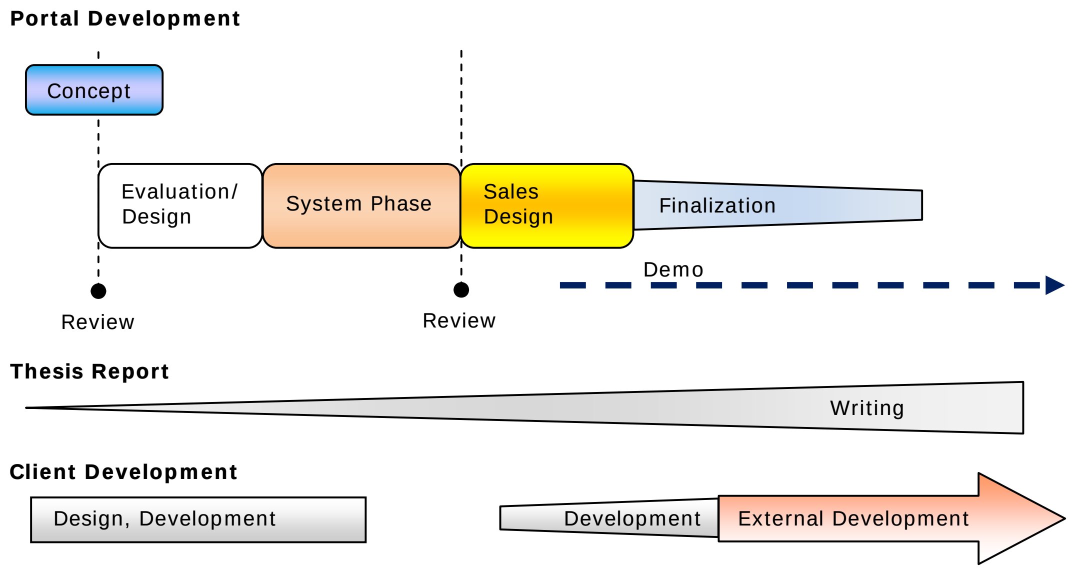


Figure 5: An overview of the time plan and the development phases.

4.2 Pre study

To get a view of which technologies that could be used in the project the internet was searched for services that handled some of the functionality that was supposed to be contained in the finished system. iTunes was one of the main idea resources, it offers about the same base functionality as this portal which is to provide the user with music, podcasts, audio books, applications and similar. Time was spent investigating if the WIP client platform should be able to connect directly to iTunes. No result came out of this more than the conclusion based on how well Apple using standards in other areas - they would probably not let you access their systems by a third party application. The iTunes application would not run on the client platform due to hardware limitations.

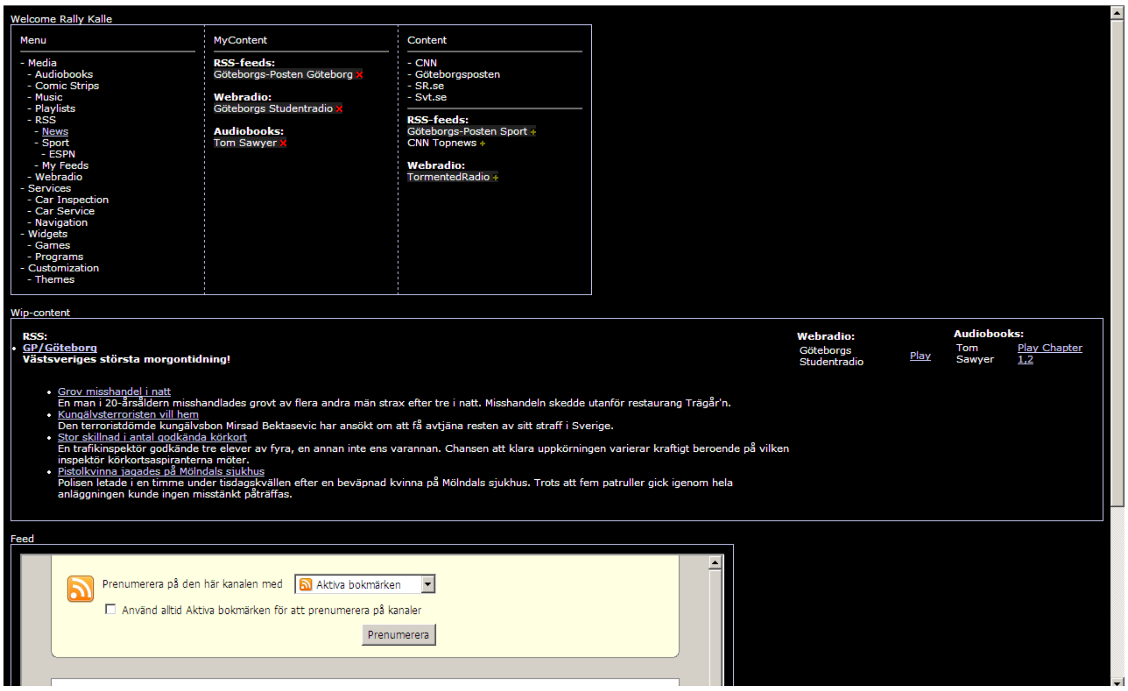
NewsML was mentioned to be used, after a short research it appeared that sites charged for news published in this format. Making of contracts for buying such content did not fit into the schedule. Another parser had also been needed and a new way of distributing the news to the client. The advantage of NewsML compared to RSS news is that the full news article could be sent to the client since the whole text is distributed in the NewsML file. News distributed with RSS is having a short summary of the article with a link to a regular webpage which in most cases does not look good on the client.

Offline mode was discussed in the pre study, if the client loses its connection towards the internet, live web radio sessions will not work especially well. Therefore the stream need to be buffered, different approaches for this could be taken. One way is that the server caches the stream and sends the parts to the client. This approach would need quite some development effort on the server system. A more simplified solution would instead be to do the buffering in the client, the problem here would be if the car is connected to internet the first part of the travel, but not before and then loses the connection, the user have to wait for the stream to be buffered to cover parts of the travel when the vehicle is not online. This would have led to silence in the beginning of the trip. The system would have been too complex if these technologies should have been implemented so they have been skipped. Ideas of having an external proxy that handled this were discussed but no more investigation in that part has been done. It was just stated as a later optional approach.

4.3 Concept

When the system environments were installed a simple implementation of the system was started, a database sketch that gave a user the possibility to select news, web radio and audio books was drawn and later implemented. Service providers were studied to see how they distributed the different media sources and a few media elements were selected and added to the first versions database. A simple interface for showing which content was available and could be selected by drag and drop was implemented. An RSS reader for showing how the data the user had selected would look like in the client was also implemented and the user were able to play audio books and web radio from the interface.

In the next meeting this functionality was shown and there was given feedback on the way it worked. After the meeting discussion between the client and server developers resulted in that the RSS format should stand as a base for the communication between the client and the server. The decision was based on the fact that much of the information that would be used in the application already was specified by this format. This means that fewer algorithms for reading different formats needed to be developed. Another aspect was that it is a waste to invent something new if there already exists something that could be used. It would also be easier for new developers to commit to the project if it is built on open and acknowledged standards. The first version of the RSS feed was plain with all elements in the base file just to test the generation of RSS feeds.



Västsveriges största morgontidning!

- Grov misshandel i natt
  En man i 20-årsåldern misshandlades grovt av flera andra män strax efter tre i natt. Misshandeln skedde utanför restaurang Trägår'n.
- Kungälvsterroristen vill hem
  Den terroristdömde kungälvsbon Mirsad Bektasevic har ansökt om att få avtjäna resten av sitt straff i Sverige.
- Stor skillnad i antal godkända körkort
  En trafikinspektör godkände tre elever av fyra, en annan inte ens varannan. Chansen att klara uppkörningen varierar kraftigt beroende på vilken inspektör körkortsaspiranterna möter.
- Pistolkvinna jagades på Mölndals sjukhus
  Polisen letade i en timme under tisdagskvällen efter en beväpnad kvinna på Mölndals sjukhus. Trots att fem patruller gick igenom hela anläggningen kunde ingen misstänkt påträffas.

☐ Använd alltid Aktiva bokmärken för att prenumerera på kanaler

Figure 6: The outcome of the concept phase, a simple menu tree, drags and drop selection of content between *Content* and *MyContent*. Preview of the selected content in the middle and the RSS feed at the bottom.

4.4 Evaluation and Design

When the concept were implemented and shown, conclusions were drawn of what drawbacks the database layout had. The concept version was in need of a new table for each new service that was added, since the final version should be able to support all kinds of different media a design that allowed this had to be created. Much knowledge from the investigation of different kinds of media was taken into thought and a specification of how the different media types should be distributed through the RSS feed was created. The following list of media types was analyzed and compared to the RSS specification [8] to see how well they fitted and which tags our RSS documents would need:

- Application
- Audio book
- Comic Strip
- Playlist
- Podcast
- Music
- News
- Settings
- Theme
- Video
- Web radio

The new distribution of the feed from the server needed to have a tree hierarchy, therefore sub feeds was needed which would take care of news, web radio, comics, and themes.

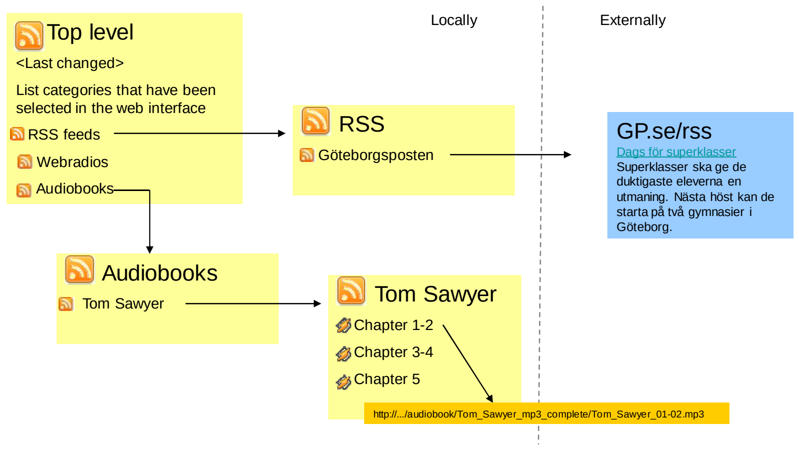

Locally
Externally
Top level
<Last changed>
List categories that have been selected in the web interface
RSS feeds
Webradios
Audiobooks
RSS
Göteborgsposten
GP.se/rss
Dags för superklasser
Superklasser ska ge de duktigaste eleverna en utmaning. Nästa höst kan de starta på två gymnasier i Göteborg.
Audiobooks
Tom Sawyer
Tom Sawyer
Chapter 1-2
Chapter 3-4
Chapter 5
http://.../audiobook/Tom_Sawyer_mp3_complete/Tom_Sawyer_01-02.mp3

Figure 7: The proposed structure of the feed that the client will subscribe to. The feed is generated based on what the user have selected in the web interface. Same type of feeds can be used for most of the services.

When evaluating the concept system and looking into the design of the real system different services on internet was investigated. This was done to see if they could be merged into the system, one example of a service that was gone through was the Sony Ericsson Play Now service. The structure of it was tracked and at first sight it appeared possible to use it in the portal or by using it directly from the client. Unfortunately the service needed an SMS confirmation to activate downloads of files so the implementation of it was stopped. A simulator of the interface was instead planned but was never implemented due to the fact that the Play Now portal with all the content was different depending on if you accessed it from a mobile phone with a valid account or with the web browser. This made it hard to reach all of its content.

4.4.1 Media content support

The following section have short explanations about what the different media types have for purpose in the system and the idea behind the selection of them. There is also a note about how they are distributed in the feed.

Application
Widgets and applications fits under the same category, the functionality to take care of a downloaded application is leaved to the client.

Audio book
A list of audio files that often is distributed in a single compressed package which made direct import of it into the system hard. In the system they will be distributed as a collection of files for the different chapters. Services for audio books exists but was only briefly investigated.

Comic Strip
A Comic Strip could work either as an RSS feed with all the strips that the client will download or as podcasts which are updated. The distribution of Comic Strips on the net varied too much for a solid solution to be selected. The comic strip idea was not part of the specification but could be seen as something nice to be met of when starting your car.

Music
Music is a main part of the media experience in a car, the idea was to offer third part media and give the user possibility to buy music.

News
News is distributed through RSS-feeds so a link to it is enough. A feed collecting all the selected news will be created.

Playlist
Decision was taken to base playlist on the open and well used format m3u [11], the playlist is a single file that links to tracks placed in the clients file system.

Podcast
A podcast is an RSS feed that links to media files, audio or video, this is one of the most important and advanced parts of the content. Since the user should be able to select individual podcasts and not download all at once the feed also needs the functionality to be integrated into the portal so each item can be selected. The customer will also subscribe to podcasts to automatically get the latest episode that is published.

Settings
The settings file is a file that should be shared between the client and the server so that changes done in the portal is set on the client and that the client can do the same to the server. For example if choosing theme in the client this should be displayed on the portal. A simple XML-file with name-value layout will be used for this purpose.

Theme
The client is built on QT and will therefore be able to use style files to customize and set the graphical interface. These files will be distributed through the portal as direct links to files. The selection of theme will be in a module that keeps tracks of different themes and their files.

Video
Videos are distributed through podcast but more services could be added for handling videos. Streaming could be one way.

Web radio
Web radio are often just links to a stream, discussion of how this should be solved in offline mode was had but ignored. In the portal the link to the stream will be published.

4.5 System Phase

The concept phase was ended by a planning and design meeting. A second version of the server system was then built. This was done from scratch but code could to some part be reused as the handling of RSS and the menu system. The ideas and conclusions from the concept phase were used to design the new system.

The system phase can be divided into a few parts, first the design phase where development between the client and server where close to each other to specify how the RSS feed should be built. Then the client developer went to vacation so the server work had to be done independently of the client functionality. After the vacation the system was reviewed and after that some smaller changes were made, mostly user interface changes but also some changes in design were made. The last phase of the system was done in part time since studies were done at the same time. This was possible since the work with the client went quite slow due to other projects having higher priority and that the web portal system was good enough for demonstration already at that time. The system got a couple of graphical changes after the demonstration.

4.5.1 Design

Based on the conclusions from the concept phase the design of the system started. First database had to be designed to handle all kinds of content that users should be able to select. The concept had different tables for each type of content, this was changed and the new database structure instead had support for every kind of media in the same table. This made the database queries a bit more complex but the support for future additions became better. Figure 8 shows how the database was designed, table names in this section is typed in italic. A table for *content* was created to handle collections of *content_data*. The *content_data* table contains every item in the system as links to music files, RSS-feeds, web radio and similar. The *content* table was created to be able to build the menu tree quickly without querying the entire *content_data* table since it contains much more elements.

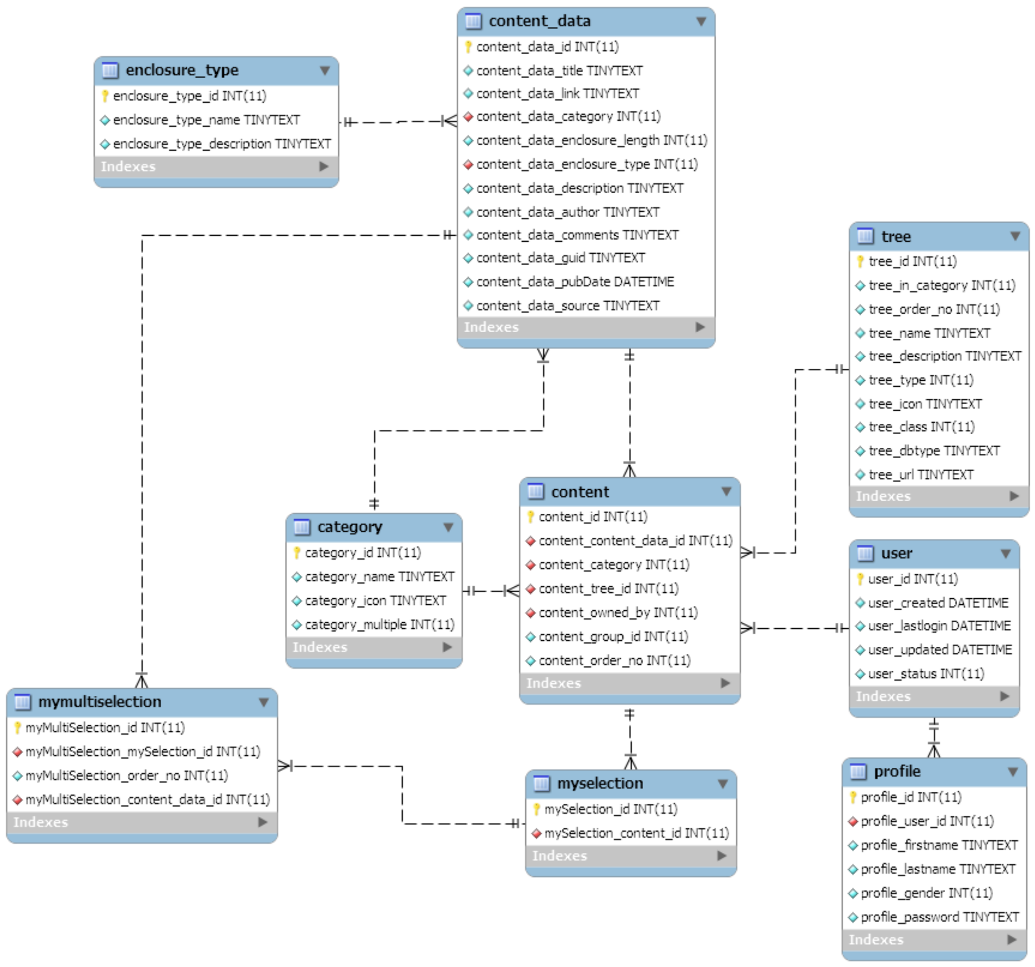
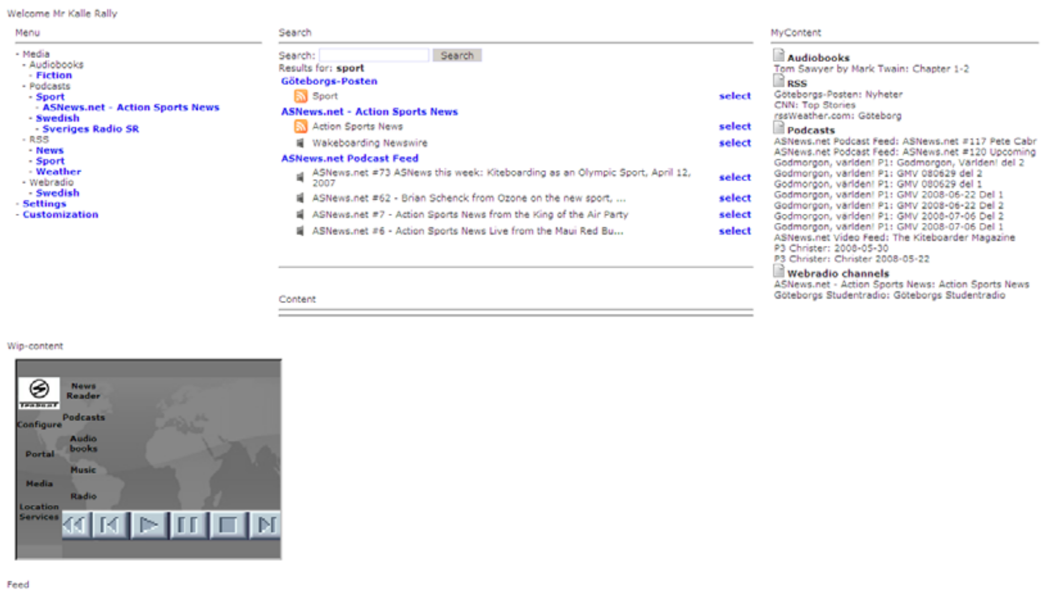

content_data
content_data_id INT(11)
content_data_title TINYTEXT
content_data_link TINYTEXT
content_data_category INT(11)
content_data_enclosure_length INT(11)
content_data_enclosure_type INT(11)
content_data_description TINYTEXT
content_data_author TINYTEXT
content_data_comments TINYTEXT
content_data_guid TINYTEXT
content_data_pubDate DATETIME
content_data_source TINYTEXT
Indexes
enclosure_type
enclosure_type_id INT(11)
enclosure_type_name TINYTEXT
enclosure_type_description TINYTEXT
Indexes
tree
tree_id INT(11)
tree_in_category INT(11)
tree_order_no INT(11)
tree_name TINYTEXT
tree_description TINYTEXT
tree_type INT(11)
tree_icon TINYTEXT
tree_class INT(11)
tree_dbtype TINYTEXT
tree_url TINYTEXT
Indexes
content
content_id INT(11)
content_content_data_id INT(11)
content_category INT(11)
content_tree_id INT(11)
content_owned_by INT(11)
content_group_id INT(11)
content_order_no INT(11)
Indexes
category
category_id INT(11)
category_name TINYTEXT
category_icon TINYTEXT
category_multiple INT(11)
Indexes
user
user_id INT(11)
user_created DATETIME
user_lastlogin DATETIME
user_updated DATETIME
user_status INT(11)
Indexes
mymultiselection
myMultiSelection_id INT(11)
myMultiSelection_mySelection_id INT(11)
myMultiSelection_order_no INT(11)
myMultiSelection_content_data_id INT(11)
Indexes
myselection
mySelection_id INT(11)
mySelection_content_id INT(11)
Indexes
profile
profile_id INT(11)
profile_user_id INT(11)
profile_firstname TINYTEXT
profile_lastname TINYTEXT
profile_gender INT(11)
profile_password TINYTEXT
Indexes

Figure 8: The database in the first version of the portal. The number of tables expanded when new functions were added to the portal but the same structure was kept.

A priority list was created to select what parts that should be implemented. First the same functionality as in the concept was implemented. Since there was a bit of complains on the colors on the concept a new simple (white) color scheme was set. A simple interface that showed a menu tree of the content categories was implemented. It let the user show and select content. A little preview window that simulated the client interface was also created. This preview window had a couple of different designs through the development due to specification changes. Finally it was rejected since it did not fit but a simple overview of the selected content was kept. One proposal was that the client application should run as a preview window, this had however needed the user to install a QT environment so it was dropped. Another idea was to have the user interface look like a car dashboard and to integrate the preview window in it; this would have taken a lot of design time and was also rejected.

A simple administration interface was created where the possibility to add content and change the menu structure. Still most insertions of content was done through hard coded

SQL statements generated by PHP scripts or by a terminal connected directly to the database server.



Welcome Mr Kalle Rally

Menu

- Media
  - Audiobooks
    - Fiction
  - Podcasts
    - Sport
      - ASNews.net - Action Sports News
    - Swedish
      - Sveriges Radio SR
  - RSS
    - News
    - Sport
    - Weather
  - Webradio
    - Swedish
- Settings
- Customization

Search

Search: Search
Results for: **sport**
Göteborgs-Posten
Sport select
ASNews.net - Action Sports News
Action Sports News select
Wakeboarding Newswire select
ASNews.net Podcast Feed
ASNews.net #73 ASNews this week: Kiteboarding as an Olympic Sport, April 12, 2007 select
ASNews.net #62 - Brian Schenck from Ozone on the new sport, ... select
ASNews.net #7 - Action Sports News from the King of the Air Party select
ASNews.net #6 - Action Sports News Live from the Maui Red Bu... select

Content

MyContent

Audiobooks
Tom Sawyer by Mark Twain: Chapter 1-2
RSS
Göteborgs-Posten: Nyheter
CNN: Top Stories
rssWeather.com: Göteborg
Podcasts
ASNews.net Podcast Feed: ASNews.net #117 Pete Cabr
ASNews.net Podcast Feed: ASNews.net #120 Upcoming
Godmorgon, världen! P1: Godmorgon, Världen! del 2
Godmorgon, världen! P1: GMV 080629 del 2
Godmorgon, världen! P1: GMV 080629 del 1
Godmorgon, världen! P1: GMV 2008-06-22 Del 1
Godmorgon, världen! P1: GMV 2008-06-22 Del 2
Godmorgon, världen! P1: GMV 2008-07-06 Del 2
Godmorgon, världen! P1: GMV 2008-07-06 Del 1
ASNews.net Video Feed: The Kiteboarder Magazine
P3 Christer: 2008-05-30
P3 Christer: Christer 2008-05-22
Webradio channels
ASNews.net - Action Sports News: Action Sports News
Göteborgs Studentradio: Göteborgs Studentradio

Wip-content

Feed

Figure 9: The first looks of the design of the system, menu tree to the left, selectable content in the middle and selected content to the right. In the bottom left the WIP preview window is shown which shows the selected content in a way that should represent how it would look on the client.

Priority was set to creating the RSS-feed the client should access for downloading the content the user had selected in the web interface; this was done as one of the first tasks after the simple user interface was created.

4.5.2 Data Collection

To test the systems capabilities a lot of data was collected from different sites. Often big chunks of data were added, for example a list of web radio channels or news feeds. To handle podcasts a module that parsed podcast feeds was created; the retrieved data was then added to the local database as content. This was needed to allow the user to select individual podcast items from a podcast. Functionality for updating this automatically was also implemented as the possibility for the user to subscribe to a podcast feed to automatically select the latest published podcast item.

When a base feed had been created the work on the portal was ahead of the client, the priorities was then moved to add more functionality on the portal as search functionality and a better administration interface.

4.5.3 Vacation Work

In the middle of the project the client developer and the salesman responsible for specifications went on vacation so the priority list was prepared with tasks that should reach the whole vacation period without need of testing towards the client. Now functionality for themes and settings was created, external services were investigated to see if they were able to be merged into the portal. Mp3.com was one service that successfully was merged into the portal, so users could select wanted tracks in the interface and the files would later be available in the feed accessible from the client. The freeware part of mp3.com was used for showing this functionality. When most of the functionality for managing media was implemented the work was directed towards the look and feel of the system. A new desktop was created to meet the user, the colors of the portal were updated and graphics was added to get a more professional feeling. Login functionality was created to make it possible for different classes of users. It also handled sessions and remembering users between sessions. Then functionality for giving car salesman the possibility to create pre defined selections of content which could be copied to newly created user. The WIP portal was updated to implement functionality for selecting content with a look and feel that should work on a tiny touch screen.

4.5.4 Review

When the vacation was over the system was shown for the steering group and a few more employees that were interested in the project. Positive feedback was given in the review. From now on most of the changes was smaller and most about the desktop page and the listing of selected content.

4.6 Sales Design

The last part of the project was to make the portal look smoother to be good enough for demonstration. The desktop was updated to be easy to use for a car salesman. The overall layout got some changes were selected theme was visible in the view were the user could see its selections. The WIP preview window was removed in this phase.

There was more merging of functions towards the client in this iteration, upload functionality was configured to handle files that the client sent to the server. On the server side there was a quite small effort to add the upload functionality but there were no specification on what the client should upload for type of data. This lead to that everything was accepted and instead some flags were set at the upload to identify if it was a file listing that were transferred or a file with settings. The idea for the upload functionality was to handle all kinds of data for future third party services. But since no specifications were done the upload functionality was kept simple but prepared for handling of files that needed parsing at time of upload.

Since there was limited time to implement new functionality some things in the user interface needed to be faked. Ideas regarding how to manage booking of services, paying for parking and downloading applications existed but no specifications or investigations had been done. These items got part of the menu of the system but there was not a backend for them. The managing of applications will however work just as a music file if applications are added to the content database. The handling of installations is up to the client to solve.

There was some changes to the profile handling late in the process, a user can be seen as a car and have multiple profiles – drivers. The change was that the different drivers should be able to select individual content and not share selections. Earlier the plan was to share most content between users and instead have a favorites section for showing the profiles preferred content. The new functionality resulted in a few database changes but this was made without very much effort even if the content selection is the main part of the portal. Instead of the favorites feed the profiles shared all the selected files but every sub feed as news or web radio was individual. If the selected files also had been individual the client would have to keep multiple file systems for each profile which probably had created duplicates in the file system, the files selected by another profile had not been selectable either. A solution could have been to tag the files with which profile that had selected them in the portal but no such functionality existed and the RSS feed format limits this with its number of tags.

There were some problems with the naming of database table users since it represented a car. There were some misunderstandings in this so registration number was added to the user database table to clearer show that it represented a car. This functionality was also needed for the car salesman interface that handles creation of users and profiles.

The WIP portal which is accessed from the client’s browser was improved in this step. It was designed to be used with a touch screen so simple one click pages was created. To maximize the user experience, pages which did not fit on the screen got buttons for scrolling up and down instead waiting for reloading of a new page with the same looks as navigating to a new page. There were some problems with the JavaScript showing popup screens. The positioning of the popup screens that was placed on a page below the first shown page did not behave as expected. The tests were done in the browser Arora which should behave similar to the clients QT platform did not interpret scripts the same way as the client’s browser which made it hard to do the testing without using the client.

4.6.1 Server

To be able to demonstrate the system, a public server was installed, GNU/Linux and Apache was used. Some problems occurred but the installation was up and running in one day. When the system was deployed on the server problems with character coding appeared and this took a while to find a solution to. The problem was that the MySQL databases together with PHP used the character set latin-1 on windows and utf-8 on GNU/Linux. When moving the data Swedish characters became corrupt. The final solution for this was to use a PHP script that collected the data from the database and converted it to the correct encoding. Quite some time was used to research how other people had done this without success.

4.6.2 Final

In the end of the project there was still planned functionality left to implement in the portal, however the system was up and running for demonstration and was shown for different companies with positive feedback. The server systems functionality was in the end of this project ahead of the client which was not ready for demonstration since it lacked graphical components.

4.7 Client Development

The development of the client went concurrent with the portal. First it was specified what functionality it needed. Then basic ideas of how the user interface should look like were drawn. The first implementation in the client was to contact the server and download the feed. Then functions for putting the downloaded files in different folders based on their stated categories were created. This became a little bit problematic regarding the sub feeds. The sub feeds was first implemented on a single file on the server identified with different query strings. This made the client to overwrite the files. Therefore the sub feeds had to be separated into different files.

When the client had got the downloading of the feed working the work on a graphical interface was started. A News reader was implemented, a media player was integrated, and support for setting different themes was created.

The development of the client later moved from the original single client developer here in Sweden to another office where multiple developers worked on the user interface for the client. This was supervised by the former developer.

5. Result

This chapter describes the functionality that was implemented in the portal; a short section will also describe the status of the client when the work stopped on the portal. The system was up and running and ready for demonstration when the project finished.

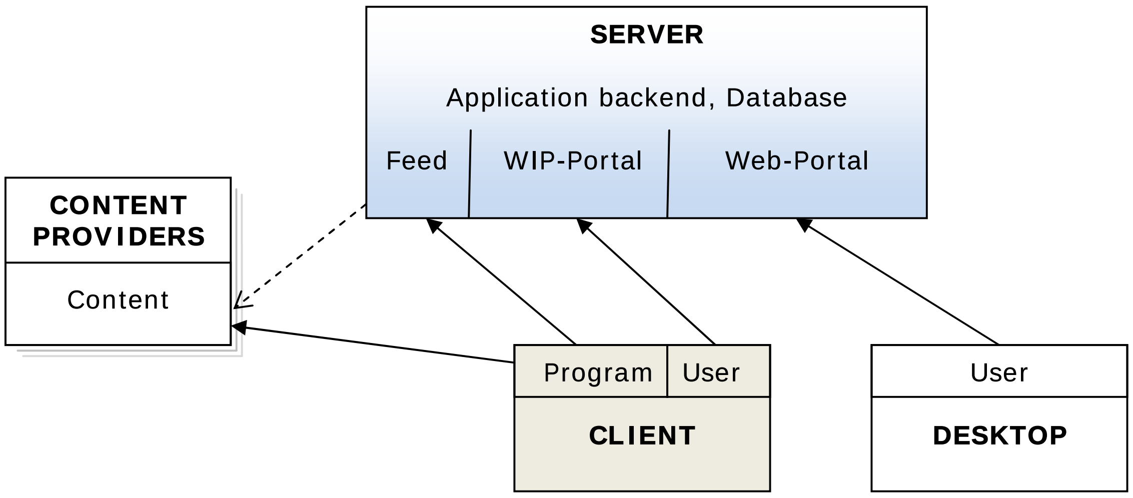


Figure 10: Showing how the different parts communicate, the dashed line means that the feed links to the content providers and that the client program is downloading the content directly from the content providers. Program means that the software is creating the request. User means that the user accesses the service manually.

5.1 The Concept

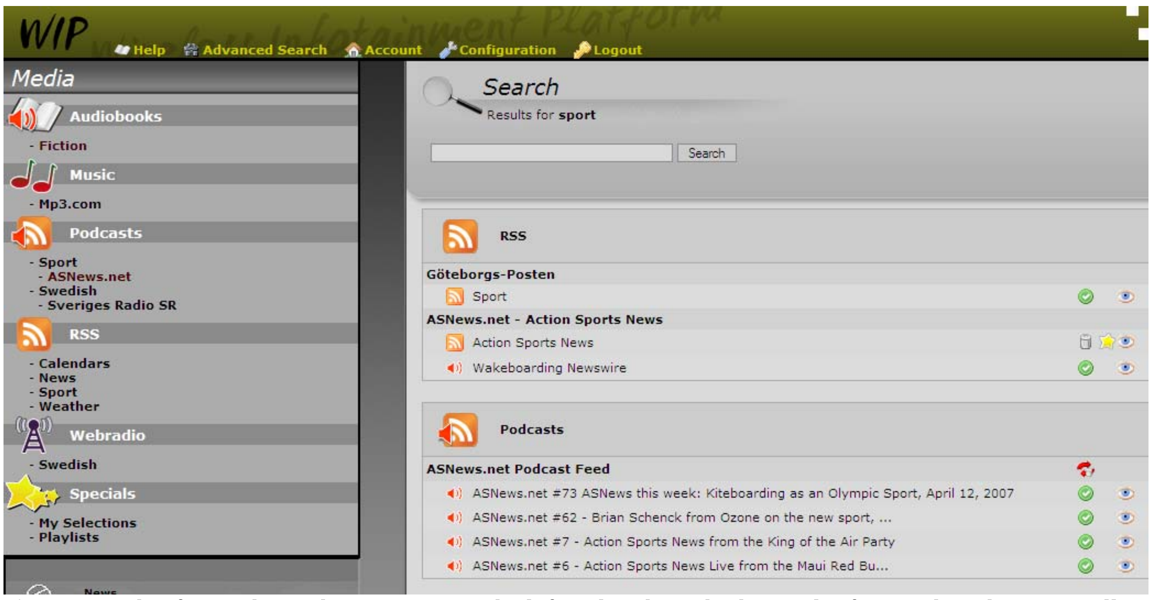
The Concept of the Infotainment Platform described a service where the user experience is simplified to selecting content in a web interface and then the media is waiting when entering the car. The goal is to have a server system that maintains content from third party providers, distributes the content to an embedded unit in a car and minimizes the effort for the end user. The user should also be able to buy content and take care of service bookings and solving payment of parking directly from the car. The interface for the user should be intuitive both on the client and portal. The client should allow the user to read news, play music, view video, listen to web radio and subscribe to podcasts and also installing applications, booking service, and paying for parking and transferring data retrieved from third party applications.

5.2 The Server System

The outcome of the project is a web portal where a user logs in and has the possibility to select different kinds of media content. The selected content can be accessed from the client through an RSS feed which links to the selected files. The system handles multiple user classes and has an administration interface; it handles sessions and can remember login information. When a user accesses the server a login screen is shown. Depending on the user class different desktop pages meets the user when logged in. A normal user is met by a list of the latest content that has been added to the system, administrators are instead met by a user configuration page where they can create new users and profiles.

To give a better overview of the system a few screenshots are helpful which will be shown later in this chapter. The system has a range of different views. The one that is most present for the user is the listing of content where content can be selected or reviewed. When selections are done the user can overview the selections in a page called My Selections.

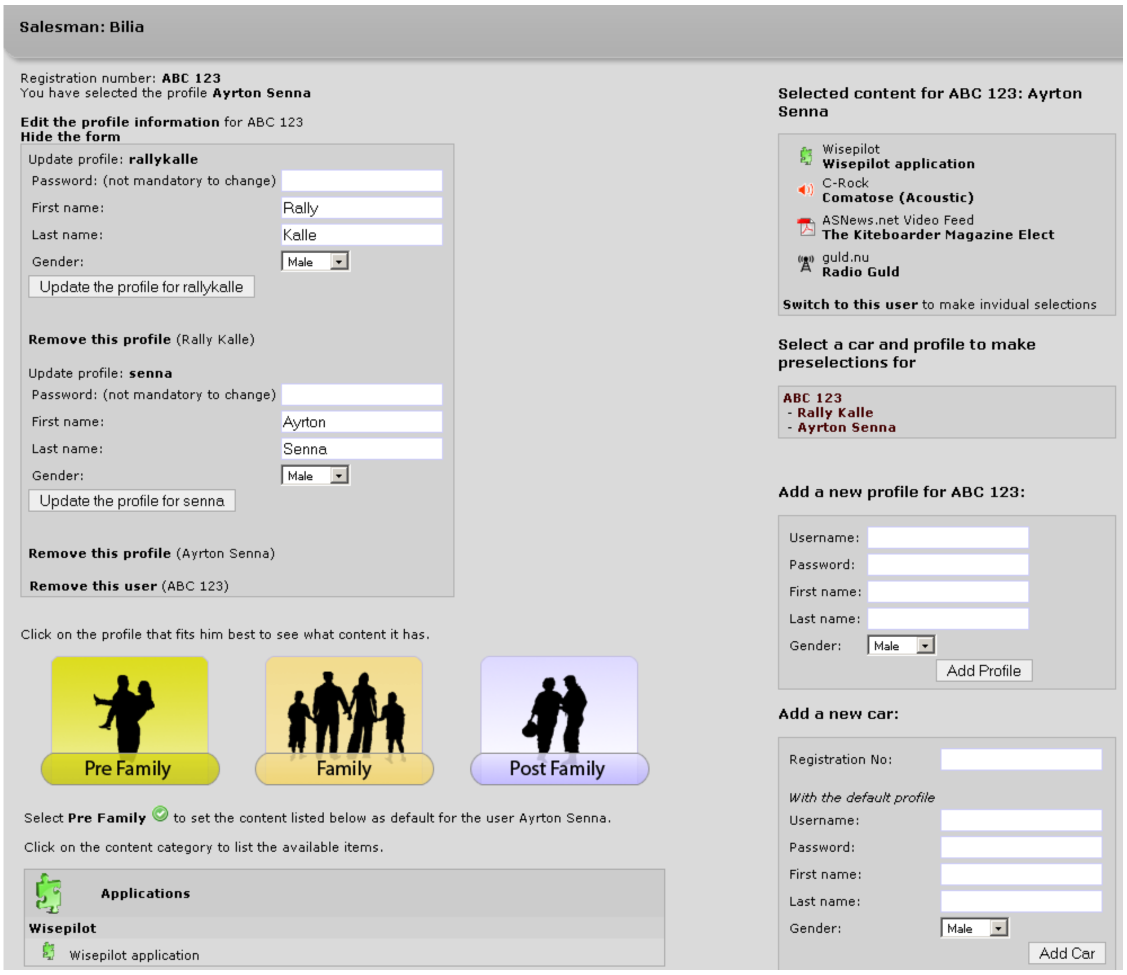
The client can upload files to the portal and when a file list is uploaded it is parsed and saved into a database, this to allow the user to create playlists in the interface. A few settings can be changed by the user; themes can be selected under configuration. The graphical interface is built on a layout where navigation between different settings and details is done in a menu placed at the top. A tree of content is present at the left and to the right the selected page is shown, figure 11 shows the layout.



Göteborgs-Posten
Sport
ASNews.net - Action Sports News
Action Sports News
Wakeboarding Newswire

ASNews.net Podcast Feed
ASNews.net #73 ASNews this week: Kiteboarding as an Olympic Sport, April 12, 2007
ASNews.net #62 - Brian Schenck from Ozone on the new sport, ...
ASNews.net #7 - Action Sports News from the King of the Air Party
ASNews.net #6 - Action Sports News Live from the Maui Red Bu...

Figure 11: This figure shows the menu tree to the left and to the right the result of a search is shown. Small buttons allow the user to select the content. For the podcast in this example the subscription can be cancelled. Each item can also be previewed.



Salesman: Bilia

Registration number: **ABC 123**
You have selected the profile **Ayrton Senna**

Edit the profile information for ABC 123
Hide the form

Update profile: **rallykalle**

Password: (not mandatory to change)	
First name:	Rally
Last name:	Kalle
Gender:	Male

Update the profile for rallykalle

Remove this profile (Rally Kalle)

Update profile: **senna**

Password: (not mandatory to change)	
First name:	Ayrton
Last name:	Senna
Gender:	Male

Update the profile for senna

Remove this profile (Ayrton Senna)

Remove this user (ABC 123)

Click on the profile that fits him best to see what content it has.

Select **Pre Family** to set the content listed below as default for the user Ayrton Senna.

Click on the content category to list the available items.

Applications

Wisepilot

Wisepilot application

Selected content for ABC 123: Ayrton Senna

Wisepilot
Wisepilot application
C-Rock
Comatose (Acoustic)
ASNews.net Video Feed
The Kiteboarder Magazine Elect
guld.nu
Radio Guld

Switch to this user to make invidual selections

Select a car and profile to make preselections for

ABC 123
- **Rally Kalle**
- **Ayrton Senna**

Add a new profile for ABC 123:

Username:	
Password:	
First name:	
Last name:	
Gender:	Male

Add Profile

Add a new car:

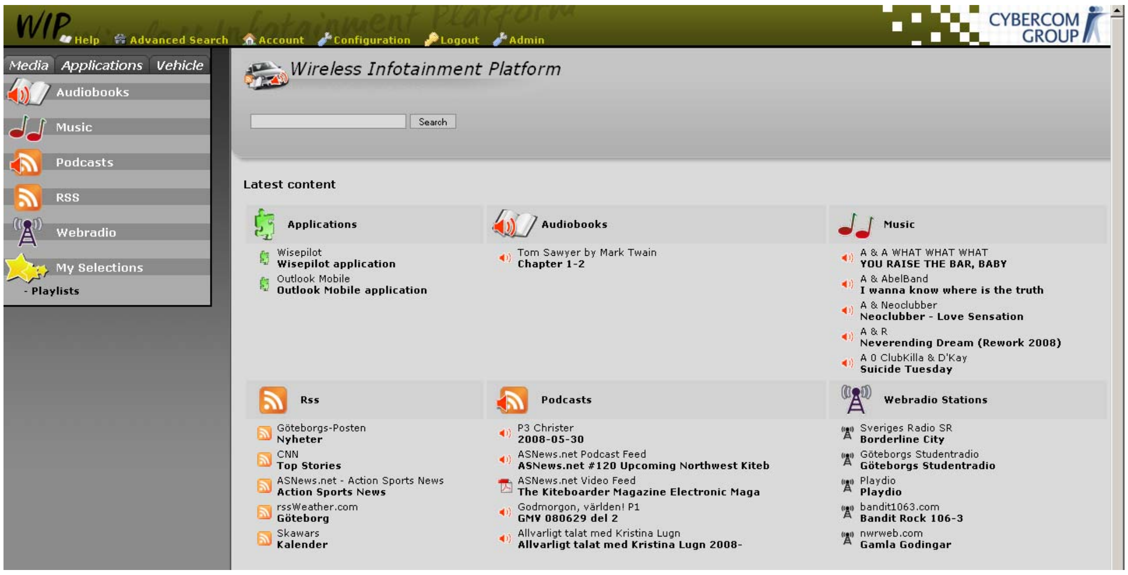
Registration No:	

With the default profile

Username:	
Password:	
First name:	
Last name:	
Gender:	Male

Add Car

Figure 12: The salesman interface where cars (users) and profiles can be added. For a selected profile pre configured content can be selected. The menu tree and the navigation menu have been cropped from this screenshot.



Latest content

Applications
Wisepilot
Wisepilot application
Outlook Mobile
Outlook Mobile application

Audiobooks
Tom Sawyer by Mark Twain
Chapter 1-2

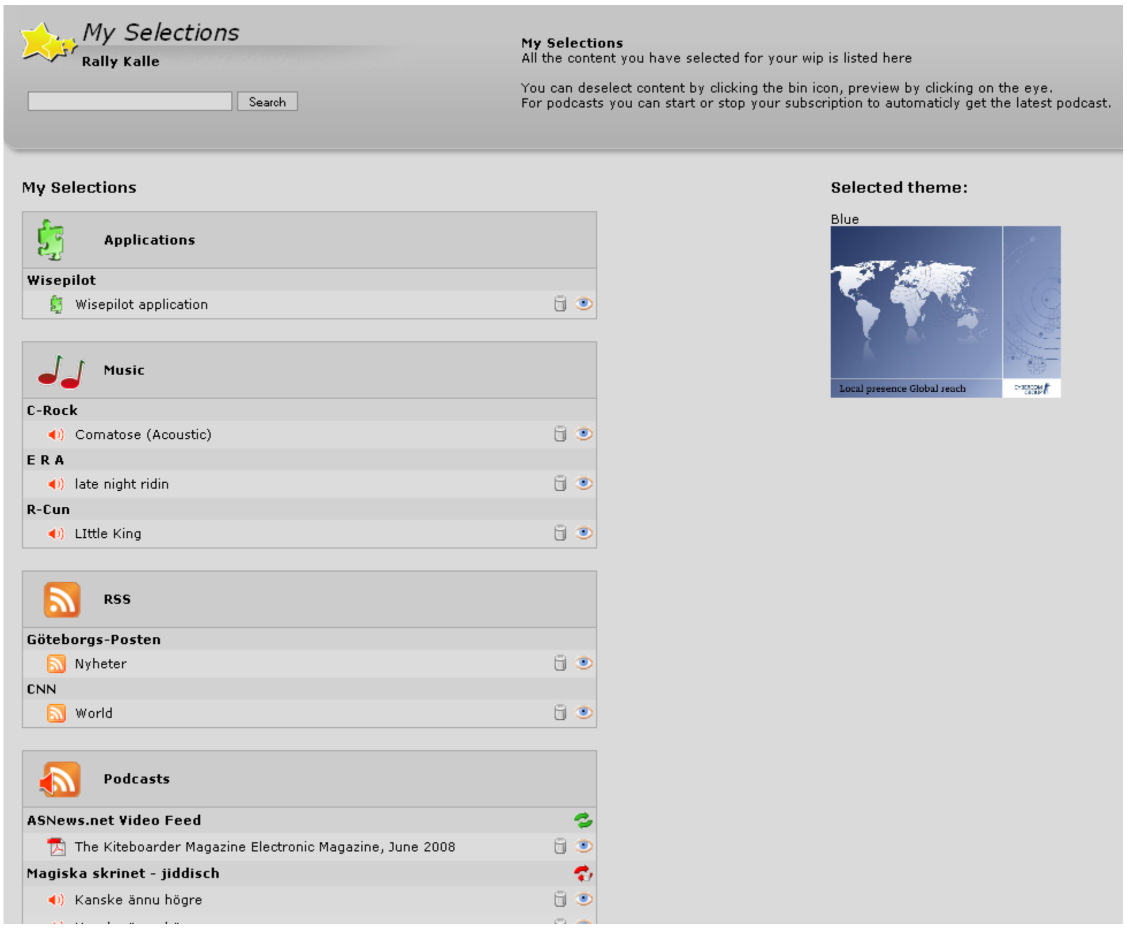
Music
A & A WHAT WHAT WHAT
YOU RAISE THE BAR, BABY
A & AbelBand
I wanna know where is the truth
A & Neoclubber
Neoclubber - Love Sensation
A & R
Neverending Dream (Rework 2008)
A 0 ClubKilla & D'Kay
Suicide Tuesday

Rss
Göteborgs-Posten
Nyheter
CNN
Top Stories
ASNews.net - Action Sports News
Action Sports News
rssWeather.com
Göteborg
Skawars
Kalender

Podcasts
P3 Christer
2008-05-30
ASNews.net Podcast Feed
ASNews.net #120 Upcoming Northwest Kiteb
ASNews.net Video Feed
The Kiteboarder Magazine Electronic Maga
Godmorgon, världen! P1
GMV 080629 del 2
Allvarligt talat med Kristina Lugn
Allvarligt talat med Kristina Lugn 2008-

Webradio Stations
Sveriges Radio SR
Borderline City
Göteborgs Studentradio
Göteborgs Studentradio
Playdio
Playdio
bandit1063.com
Bandit Rock 106-3
nwrweb.com
Gamla Godingar

Figure 13: The desktop, the page a normal user sees when logging in. The latest content that is added in each category is listed.



My Selections
All the content you have selected for your wip is listed here

You can deselect content by clicking the bin icon, preview by clicking on the eye.
For podcasts you can start or stop your subscription to automaticly get the latest podcast.

My Selections

Applications
Wisepilot
Wisepilot application

Music
C-Rock
Comatose (Acoustic)
E R A
late night ridin
R-Cun
LIttle King

RSS
Göteborgs-Posten
Nyheter
CNN
World

Podcasts
ASNews.net Video Feed
The Kiteboarder Magazine Electronic Magazine, June 2008
Magiska skrinet - jiddisch
Kanske ännu högre

Selected theme:
Blue

Figure 14: My Selections page showing the content the user have selected, icons for removing selections, previewing the content and subscribing to podcasts is listed. The selected theme is shown to the right. This picture has been cropped.

5.2.1 Content

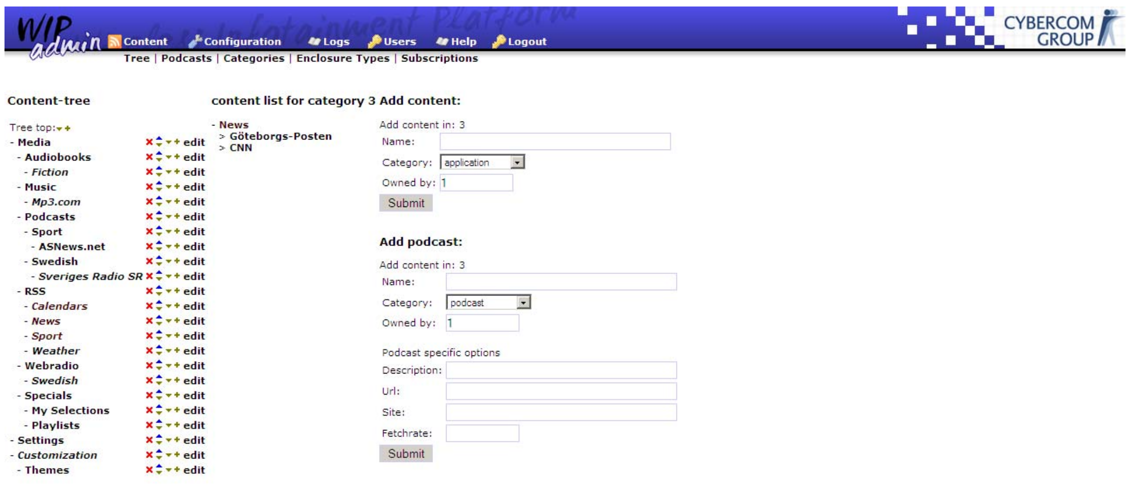
The idea of the portal is that all kinds of content should be possible to add since the portal is only linking to the content. Some content is however more advanced and special functions have been added for them. For example podcasts which are parsed remotely and added to the system so that the user can select individual files or start a subscription that automatically selects the latest available podcast when it is published. The fetching of podcasts is scripted in the backend and can be scheduled.

A menu tree is visible that gives the user the possibility to navigate to different content. Icons have been added to the menu tree to easier identify each category.

Search functionality has been added to let the user find content placed in the database. The database currently contains 529 groups of content (content table) with a total of 11573 items (content_data table).

5.2.2 Administration Interface

The administration interface has most of the functionality for configuring the different parts of the system. The interface is not very user friendly and not deployable to the end market. The system is built for functionality, not ease to use. The interface offers an administrator to edit the menu tree, add content to the different categories. Podcast has an own form for adding items since it should be added to the podcast parser engine also. Added podcasts can later be refreshed through the interface to get the latest content from the remote site. The podcast service connects to a remote RSS-feed, parses it and podcast items that not already exist in the database will be added. User subscriptions of podcasts can be viewed in the interface. Update of the music library that is connected to mp3.com can also be done.



Content-tree

Tree top:

- Media edit
  - Audiobooks edit
    - *Fiction* edit
  - Music edit
    - *Mp3.com* edit
  - Podcasts edit
    - Sport edit
      - ASNews.net edit
    - Swedish edit
      - *Sveriges Radio SR* edit
  - RSS edit
    - *Calendars* edit
    - *News* edit
    - *Sport* edit
    - *Weather* edit
  - Webradio edit
    - *Swedish* edit
  - Specials edit
    - My Selections edit
    - Playlists edit
- Settings edit
- *Customization* edit
  - Themes edit

content list for category 3

- News
  > Göteborgs-Posten
  > CNN

Add content:

Add content in: 3
Name:
Category: application
Owned by: 1

Add podcast:

Add content in: 3
Name:
Category: podcast
Owned by: 1

Podcast specific options
Description:
Url:
Site:
Fetchrate:

Figure 15: Screenshot of the administration interface, here showing the editing of the menu tree.

5.2.3 Modules

To distinguish from the main part of the portal some functions are added aside in the code and called modules. For example the Podcast parser which reads remote servers feeds and collecting links to all episodes to make them selectable of the user. Another module is the parser for mp3.com; this was only used to go through all their pages to retrieve the free music once. It can however be used again to discover newly added tracks and duplicates will not occur due to the coding functionality. The result from this parser was that the portal could show how a lot of music files could be distributed.

5.2.4 Users

The portal is a multiuser system; this means that there are different classes of users with different possibilities and the portals look and feel changes depending on what user that has logged in. The following user classes are present:

Normal User
The normal user is connected to a car; multiple profiles can be bound to one user which all has different usernames and passwords. The profiles will share the selected content that need to be downloaded. Smaller entries as themes, news and web radios are individual.

Administrator
The administrator's role is to add content; they can update podcasts and edit the menu tree. They can also edit users, create settings and check logs.

Salesman
To simulate an interface for a car salesman that uses the portal from the store a user of that class exists. It has the possibility to create cars (users) and add profiles to them. Then a profile with pre selected content can be copied to the created user.

Pre-defined
This user class is a container for selected content; they appear as a normal user in the system and can select content. Then a Salesman can copy the profile to a newly created user instead of selecting all content again. There are three default profiles; Pre-Family, Family, Post-Family. The names are based on how Volvo addresses their target market segments.

5.2.5 API

The communication towards the client is made in both directions; most work was done in the feed that the client reads. The feed is built with XML, with the RSS specification as a base for the structure. All content that the user has selected is presented in this feed, it its layered in a tree structure with sub feeds that the client can download and keep without changing them. One example of this is the web radio feed where the sub feed file contains links to the selected web radio channels.

```
<item>
  <title>Settings</title>
  <link><![CDATA[http://xxxx/wip/v2/feed_settings.php]]></link>
  <description></description>
  <author>auth</author>
  <category>settings</category>
  <guid isPermaLink="false">settings12345</guid>
  <pubDate>Wed, 12 Nov 2008 12:58:38 GMT</pubDate>
  <comments></comments>
  <source></source>
</item>
```

Figure 16: Example of what content that is present in the XML feed. Here the item links to the settings.

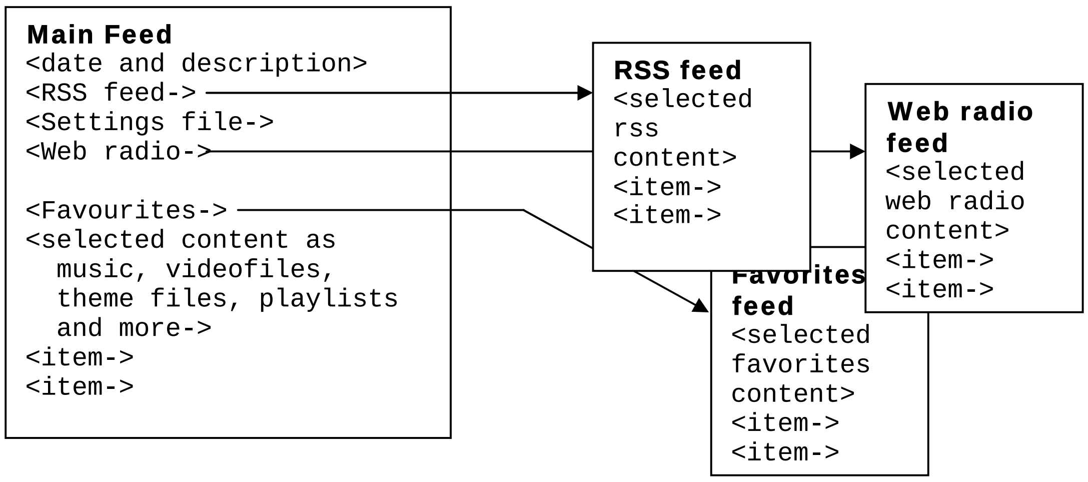


Figure 17: The feed is build like this; the main feed links to sub feeds for content that the client not will download at once. Settings are a link to an XML-file. Items in the bottom are all links to content that the user have selected.

The client can upload files to the server; three different types of files are treated in special ways. When the client uploads a file it specifies which type it is. The following list describes the types:

Error report: *"Error"*
These files are renamed and sorted by user and profile id to make overview of errors easier.

Configuration file: *"Settings"*
The settings file, which is stored in XML format, is parsed and settings for the user are updated on the server.

Listing of client file structure: *"File list"*

An RSS file with all the files the client has on its file system, these are added to a file list database on the server so the user can create playlists based on the content. When a file is added to the database the date is stored, this gives the possibility to remove files from the file list database when they do no longer exist on the client. The reason for not doing this automatically is if the client have had an external drive with music that might be connected again, a playlist using these files should not be corrupt just because of that the files are not present at the time of upload. This functionality could be improved further; playlist was just added as proof of concept.

Playlists are files that keep file paths to media on the client; the m3u-format [11] is used for the playlists.

5.2.6 Code

The coding has been done in PHP, HTML and JavaScript, mostly a fairly plain structure have been used with the code divided into different directories and functions. The code reaches 8865 lines separated into 82 files. External open source libraries have been used for JavaScript.

The file structure is as following, public files are stored in the root directory, graphics, JavaScripts and included PHP-files are stored in separate subdirectories. Files handling XML and SQL queries are also in separate directories, the SQL files have also got the prefix *sql_* to easier distinguish them from files with the same name that calls them and generates the html code. Modules are in separate subdirectories. The access from the web browser is pointed to one of the files in the root directory which includes configuration files to minimize the duplication of code. Layout is generated by a single document to keep it consistent.

All actions where user input update the content, like selections or adding or removing entries in the database has been collected into one file. This to minimize the input and output in different files to avoid security leaks.

5.2.7 Database

The database is stored in MySQL; most of the data in the portal is received from the database and not hardcoded in the PHP-files. Multilingual support was not specified so strings are hardcoded. Figure 18 shows the database structure, it also summarizes the functionality that can be found in the system.

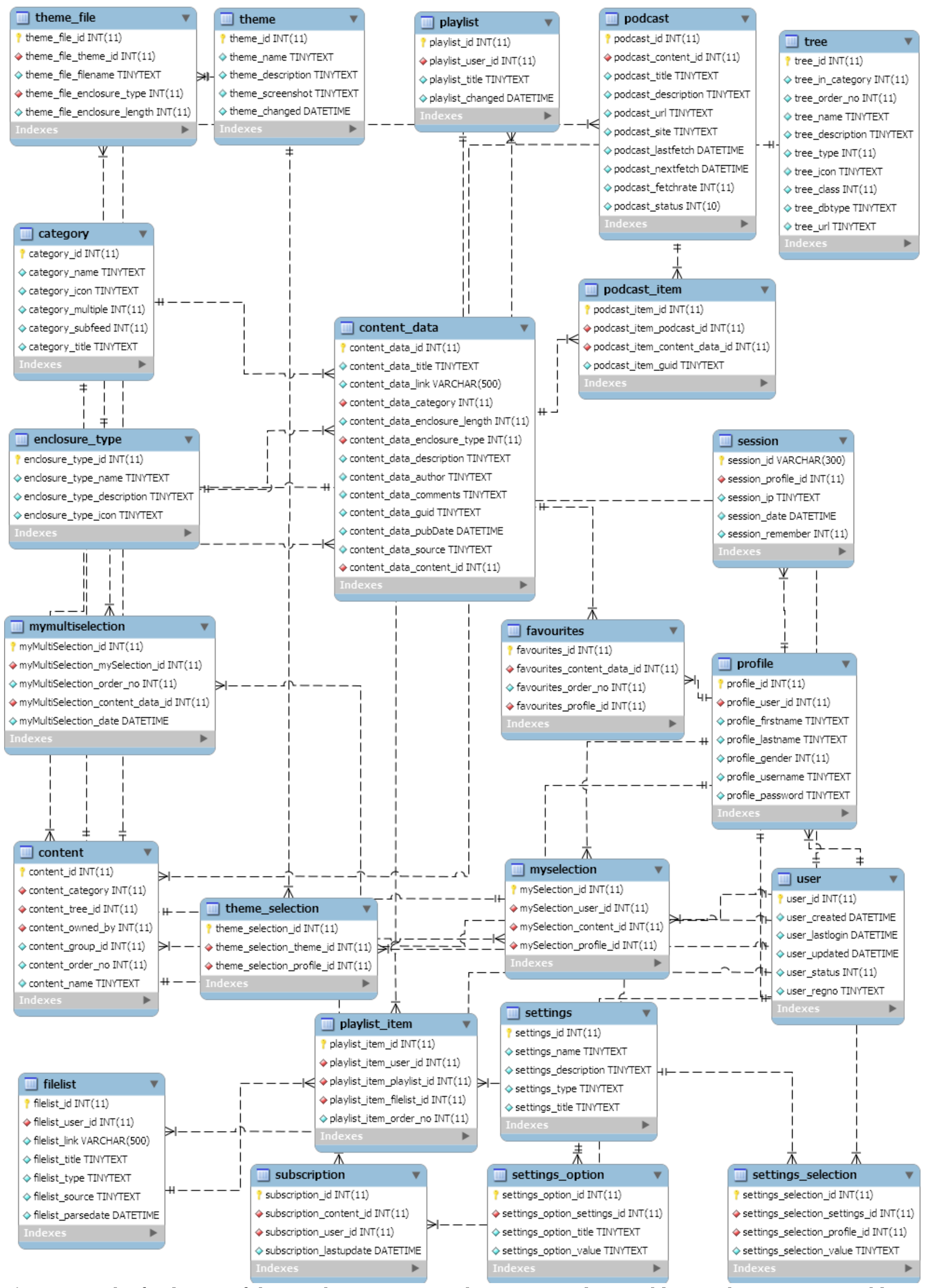

theme_file
theme_file_id INT(11)
theme_file_theme_id INT(11)
theme_file_filename TINYTEXT
theme_file_enclosure_type INT(11)
theme_file_enclosure_length INT(11)
Indexes
theme
theme_id INT(11)
theme_name TINYTEXT
theme_description TINYTEXT
theme_screenshot TINYTEXT
theme_changed DATETIME
Indexes
playlist
playlist_id INT(11)
playlist_user_id INT(11)
playlist_title TINYTEXT
playlist_changed DATETIME
Indexes
podcast
podcast_id INT(11)
podcast_content_id INT(11)
podcast_title TINYTEXT
podcast_description TINYTEXT
podcast_url TINYTEXT
podcast_site TINYTEXT
podcast_lastfetch DATETIME
podcast_nextfetch DATETIME
podcast_fetchrate INT(11)
podcast_status INT(10)
Indexes
tree
tree_id INT(11)
tree_in_category INT(11)
tree_order_no INT(11)
tree_name TINYTEXT
tree_description TINYTEXT
tree_type INT(11)
tree_icon TINYTEXT
tree_class INT(11)
tree_dbtype TINYTEXT
tree_url TINYTEXT
Indexes
category
category_id INT(11)
category_name TINYTEXT
category_icon TINYTEXT
category_multiple INT(11)
category_subfeed INT(11)
category_title TINYTEXT
Indexes
podcast_item
podcast_item_id INT(11)
podcast_item_podcast_id INT(11)
podcast_item_content_data_id INT(11)
podcast_item_guid TINYTEXT
Indexes
content_data
content_data_id INT(11)
content_data_title TINYTEXT
content_data_link VARCHAR(500)
content_data_category INT(11)
content_data_enclosure_length INT(11)
content_data_enclosure_type INT(11)
content_data_description TINYTEXT
content_data_author TINYTEXT
content_data_comments TINYTEXT
content_data_guid TINYTEXT
content_data_pubDate DATETIME
content_data_source TINYTEXT
content_data_content_id INT(11)
Indexes
enclosure_type
enclosure_type_id INT(11)
enclosure_type_name TINYTEXT
enclosure_type_description TINYTEXT
enclosure_type_icon TINYTEXT
Indexes
session
session_id VARCHAR(300)
session_profile_id INT(11)
session_ip TINYTEXT
session_date DATETIME
session_remember INT(11)
Indexes
mymultiselection
myMultiSelection_id INT(11)
myMultiSelection_mySelection_id INT(11)
myMultiSelection_order_no INT(11)
myMultiSelection_content_data_id INT(11)
myMultiSelection_date DATETIME
Indexes
favourites
favourites_id INT(11)
favourites_content_data_id INT(11)
favourites_order_no INT(11)
favourites_profile_id INT(11)
Indexes
profile
profile_id INT(11)
profile_user_id INT(11)
profile_firstname TINYTEXT
profile_lastname TINYTEXT
profile_gender INT(11)
profile_username TINYTEXT
profile_password TINYTEXT
Indexes
content
content_id INT(11)
content_category INT(11)
content_tree_id INT(11)
content_owned_by INT(11)
content_group_id INT(11)
content_order_no INT(11)
content_name TINYTEXT
Indexes
theme_selection
theme_selection_id INT(11)
theme_selection_theme_id INT(11)
theme_selection_profile_id INT(11)
Indexes
myselection
mySelection_id INT(11)
mySelection_user_id INT(11)
mySelection_content_id INT(11)
mySelection_profile_id INT(11)
Indexes
user
user_id INT(11)
user_created DATETIME
user_lastlogin DATETIME
user_updated DATETIME
user_status INT(11)
user_regno TINYTEXT
Indexes
playlist_item
playlist_item_id INT(11)
playlist_item_user_id INT(11)
playlist_item_playlist_id INT(11)
playlist_item_filelist_id INT(11)
playlist_item_order_no INT(11)
Indexes
settings
settings_id INT(11)
settings_name TINYTEXT
settings_description TINYTEXT
settings_type TINYTEXT
settings_title TINYTEXT
Indexes
filelist
filelist_id INT(11)
filelist_user_id INT(11)
filelist_link VARCHAR(500)
filelist_title TINYTEXT
filelist_type TINYTEXT
filelist_source TINYTEXT
filelist_parsedate DATETIME
Indexes
subscription
subscription_id INT(11)
subscription_content_id INT(11)
subscription_user_id INT(11)
subscription_lastupdate DATETIME
Indexes
settings_option
settings_option_id INT(11)
settings_option_settings_id INT(11)
settings_option_title TINYTEXT
settings_option_value TINYTEXT
Indexes
settings_selection
settings_selection_id INT(11)
settings_selection_settings_id INT(11)
settings_selection_profile_id INT(11)
settings_selection_value TINYTEXT
Indexes

Figure 18: The final view of the Database structure. The content and user tables are the center part, tables for settings and theme is added on top. The arrows show how tables are related to each other.

5.2.8 Surveillance

The system is logging failed SQL queries and other faults that might be generated; this is stored in a database and can be accessed from the administration interface. The code offers to set the system in debug mode to ease development and find errors quicker.

5.2.9 Missing Features

The portal project does not currently have an end, as much functionality as possible was supposed to be implemented and the limited time and then other resources will continue. Here follows a list of tasks that were never crossed of the priority-list.

- New expanding listing interface for My Selections
- Theme of the portal following client theme
- Wizard for pre selections
- Improved listing of directories
- Feed for setting files that the client should download and upload
- Scheduling for podcast fetching
- Expanded search functionality as saving of searches
- Improved desktop

The features listed above can be seen as minor. Some more functionality on the file sharing between the client and server is needed to handle third party applications that should handle data that is sent from the client. For example functions like diagnostics of the car but this have not been set as a requirement.

5.2.10 The WIP Portal

The WIP Portal was created mostly as a proof of concept where the user from the client should be able to access a simplified part of the portal. The interface is built with the idea that it should be easy to navigate with a touch screen. The WIP portal has only the functionality to select content. Pages for other functionality as configuration, settings theme and showing the currently selected content have been created as placeholders for a better overall view but do not contain any functionality.

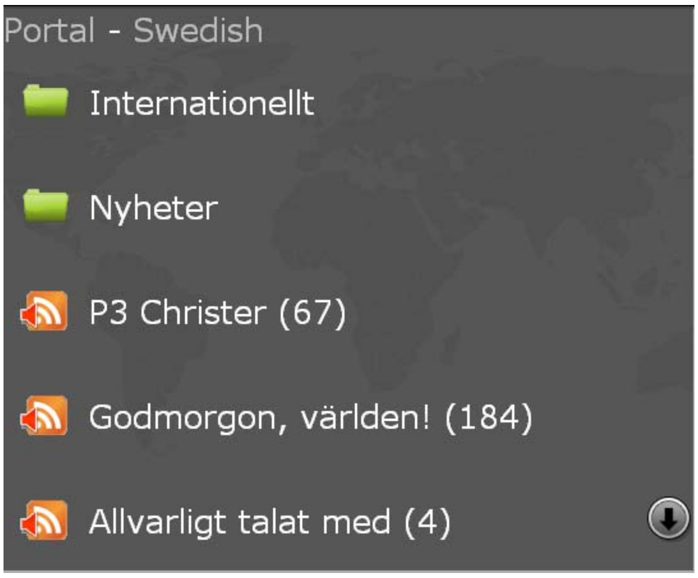


Figure 19: The WIP Portal, accessible from the client's browser. Scrolling is done by clicking the arrow.

5.3 The Client

The client is outside the scope of this report but to get a better view of the project some basic details will be described, the client has been developed concurrently with the portal and both the projects have affected each other.

The client is running on a Linux platform using a tiny touch screen; the target for it is the automotive industry with embedding in cars. The navigation of the user interface of the client is made to be useful when the user sit at the driver's seat. The client can play music, web radio and video. It has a functionality for reading RSS feeds and display the news as articles. To access these files the client connects to the portal and retrieves the list of files it should download, these are then downloaded if they are changed later than last fetch or do not exist in the client. The client is fetching the RSS feed from the server over and over again when it is online to see if changes have been made.

The client has also a web browser that gives it possibility to access the WIP portal on the server to select or deselect content in a very simple interface.

The hardware in the client has the possibility to connect to internet over WLAN or through a mobile connection. It has USB input ports for handling external devices as memories and hard drives.

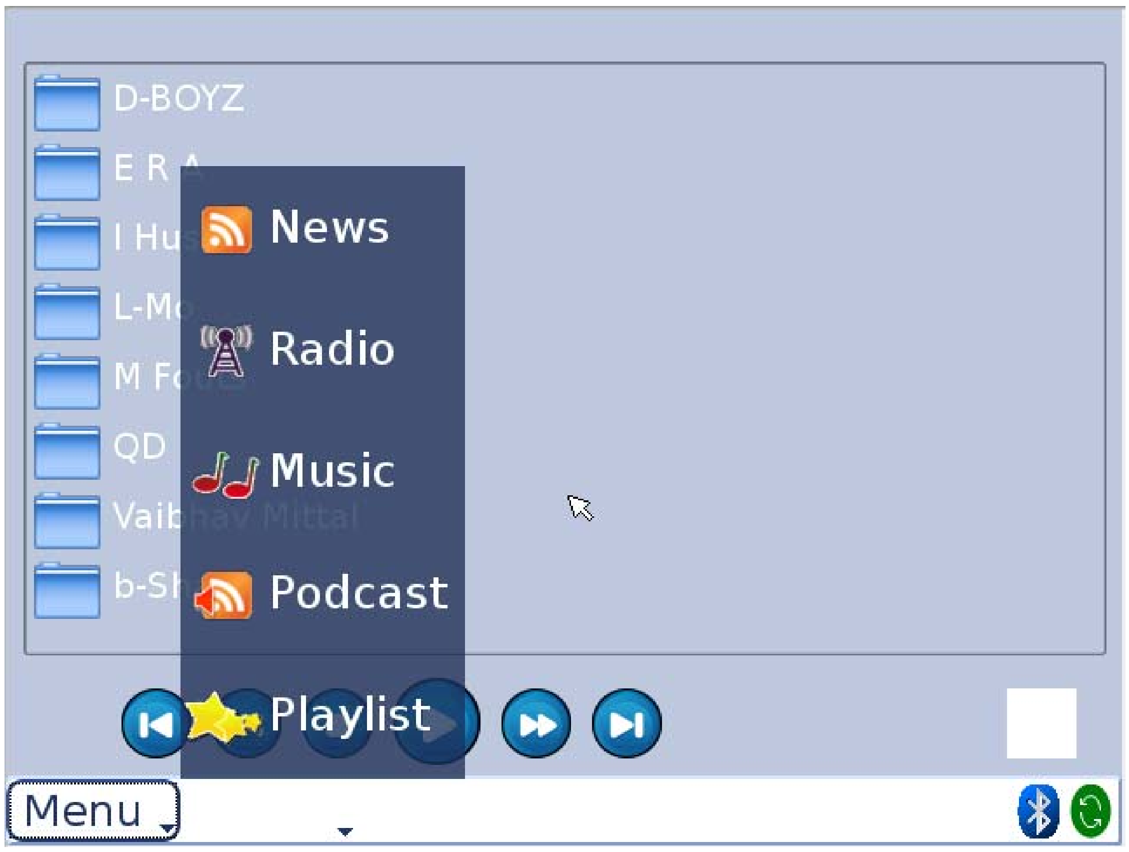


Figure 20: This is the client user interface; in the background listing of music can be seen together with the media player. In the foreground the menu for navigation between the different services is shown, the icons were created for the portal but have been reused in the client.

6. Discussion

This chapter will discuss the decisions made in the project, why they were made and what good outcome they gave but also what could have been done differently to achieve a better result.

6.1 Time Plan

The overall time plan for the project was set to handle breaks and cover for problems since it was planned over far more than the 20 weeks the total work on the project and thesis was supposed to take. No major problems occurred in the project that caused work to be thrown away or redone. Instead other work came in the way and the project got lower priority. When the deadline got closer the work on the thesis instead had to be maximized to pass the deadline criteria.

6.2 Method

The project started with a couple of power point presentations that described an idea of functionality. Then the idea needed to be applied on systems to see what was possible to implement in the short time this project was planned to be finished. Due to this short time from idea to system less time was spent on creating specifications, instead the work flew forward and specifications was made when they were needed. This was up to the developers to cope with; the problem that occurred here was that the picture of what the system actually should do or how it should be shown was not clear. Neither the audience for the demonstration was set. This made it a little bit hard to decide what should get the highest priority and how the graphical design choices should be done. Therefore the systems functionality had to be extensive to be prepared for all kinds of designs.

Better specifications could probably have made the work easier later on; it is still not really decided how the user management should work with profiles and different cars or how the client handles login to the portal. It was a problem in the project that some functionality for the client did not get decided so that the portal could not implement those parts. There was not a technical project leader in the project that made clear decisions of what that should be done; more steering of the project is something that could have improved the quality of the implementation

A more waterfall like method had made clearer specifications of what that should be done, and made the actual work easier. But since there was little knowledge of how things would end up, the first ideas was far from technical and there was really a need of an hands on example to clear some things of what was possible to implement. Some examples is how the content should be managed, if all content should be moved to the company server, this would later have needed a large effort in maintaining content and a large server farm that could handle all data. Other parts were which services that actually could be implemented, there were purposes on integrating both iTunes and the Sony Ericsson Play Now but none of these services was open enough to be usable.

The concept phase with the prototype that was thrown away gave good insight on what things that was hard to implement. It was quickly implemented since the design of it would not be used later so therefore some assumptions and quick solutions could be used instead.

There were some problems in the project with file sharing and that the development of the portal was done on a laptop, a global file area should have been setup but this was never done. There was therefore no repository to keep track of changes in the code. When the server finally got installed, a Subversion [31] repository for version control was setup. There were problems with the user login handling in it so it was deprecated again when the server moved to a public location. Since it was only one developer of each part of the client and server, version management was not so important but it had been easier to track the changes to get a better overview of what was implemented. Tools for priority list and bug reporting should have been good to get a more formal view of additions to the system.

6.3 Design

The design of the systems was mostly carried out by the two developers; a more controlled approach could have affected the system in both a good and bad way. The development had probably taken more time but more formal guidelines might have given the project more structure.

6.3.1 Solid Structure

Since the portal system was not supposed to be in a production environment much cheating could probably have been done just to have a nice front end without real functionality. However an earlier experience where the development of a test prototype became a part in a production system made the ground for building a solid structure that the portal could rest on. Afterwards it can be seen that the user functionality was a bit too advanced with sessions and multiuser system since the demonstration just showed some more or less hardcoded examples of user administration. The advantage with this advanced user functionality that was created in the beginning gave the possibility to create the pre defined users and add the salesman user without redesign. Late changes of how the selections between users and their different profiles became a minor change since the database structure had most of the functionality to handle different profiles even if it was not specified how it should be done.

Some drawbacks could be seen in the database structure, the work went slower and slower when the queries became more and more complex to include all the added functionality and to handle different kinds of selections. The structure was however maintainable and was open for changes to handle the design changes late in the project. More time probably should have been spent in the design phase of the database. The idea of the design was that the content should be divided into two tables; one that collects groups of content and one with the content. The problem that occurred with this was that the selection should be done on both the content and the group. This made the queries a bit more complex than they had been otherwise. The reason for having the grouped content table was to minimize the need of doing queries of the content data table when rendering the menu tree since it contained more than ten times the number of rows than the content table.

6.3.2 Language Choices

The scripting language PHP was chosen since knowledge in that language was good and it is useful for rapid development since there are not a lot of constraints or typing. The drawback with PHP is the debugging since you do not need to use types for variables, which have been taken advantage of in this project. Not so much time have been spent on correcting bugs so the language selection can be seen as successful. Some bugs appeared when changing the way selections were done regarding user and profile id. Since no compiler is going through and checks that the correct parameters are sent into functions problems did occur when this was changed. The technical reason was that the profile id needed to be sent into all the updated methods. Some of these methods did not get updated and this was only discovered when running the application and executing certain tasks.

Earlier projects that have been done in the university studies have used Java and frameworks that generate the HTML code. Had these methods been used instead had the work probably taken a lot more time. Two tools that have been used in an earlier school project are Tapestry and Cayenne. Tapestry is used to generate dynamic web pages from Java code [17]. Cayenne is a tool for creating database connections and database tables for Java code [16]. When these tools were used they created code that did not make sense and database changes were very problematic since Cayenne often needed reverse engineering of the database creation to modify the databases. The databases in this project were created by queries written in the PHP code, and then the tables were filled with data from PHP scripts, such insertions had definitely gone slower with another language.

A Java Servlet structure would in some cases have had advantages of the PHP based design. For example the way the client connects to the server to get updated feeds. Instead of making a lot of upstream connections a keep-alive connection could have been used instead to inform the client that a change on the portal has been done. Thereafter the client could have received the whole feed. A keep-alive connection could however be problematic if the client is not online all the time and looses the connection to the server. Another aspect where Java would have worked better is the podcast fetching. As implemented now a cron job together with a shell script have to activate a PHP page multiple times to go through all elements that is stored in the podcast selection. If solved by Java or another language which makes an executable that runs without timeouts this program could have been run once and automatically quit when finished. The same is now done with PHP but not in as an elegant way. One reason why PHP was used for this fetching and not adding another language was to minimize the applications and environments needed for the system to run.

The client interface was built on QT. This framework has the possibility for adding c-code into the built in browser to handle normal HTML-tags differently. This could have been used for building the full user interface on the server and then let the client be a thin browser. This approach was discarded since there had been problems with the client working in offline mode and when working on the local file system it could be harder for the server to generate correct code. Integration with a media player would also become a more difficult task. There would also be security problems, if a certain tag made the client to list its file system and integrated that into the web page, navigating to an external web page with the same tag would open for exploits. Exactly if this could be done or not is unclear since this road was not taken. The advantage that could have been achieved with the server generated interface is that it could be changed quickly.

6.3.3 Building from scratch

The portal has been built from scratch, instead of building on some existing platform, the reason for this is also based on earlier experience, from OsCommerce [29] and phpBB [30] which are two open source PHP projects with complex codebases. OsCommerce is a web shop and the code for it is not well structured and design changes make a lot of duplications in the code. The other example taken here where earlier code experience exists is phpBB which is a forum. It has a better built codebase than OsCommerce but have drawbacks when bigger design changes need to be done. It is built in a certain way and would not work in another project without large changes that would take a lot of time. These two projects are taken as examples since there existed experience from them; they are not applicable on this project since it is so different. They are taken up to show that even minor changes in existing projects takes a lot of time and it can therefore be quicker to invent the wheel again. Some parts of the portal are however built after checking examples of how things usually are done to make them safe and stable, for example the login and user management. The parsing and generation of RSS feeds have got inspiration from open source libraries. These functions are however minor in the portal and most of the work have been spend on the selection of content and the insertion of content in the system. This had not been as easy building on something existing. Another aspect of building from scratch is that since this was a one developer project, changes in a system you know everything about goes a lot quicker than editing someone else’s code in which you are not aware of side effects. The first functions that were added to the system were not standard functions and therefore building from scratch seemed like the only option.

6.3.4 Drawbacks with RSS

Some minor problems occurred by using RSS but most of them could be worked around. The RSS format is limiting since there are not so many tags, some tags was therefore used to other purposes than stated in the RSS specification. When studying podcast feeds on the net it appeared that iTunes had some specific tags in the feeds [13] so the decision was made that it was alright to modify the RSS scheme a little bit.

When creating the feed that should connect the client and the server there was some problems with how dates should be handled. The question appeared whether if files should be updated or not and what the date should represent. Either if when the file was added or when it was updated. The agreed specification did not specify this and the opinion from the client and server developers were different which resulted in some bugs before the specification was clear.

6.4 Result

The result of the project can be seen as successful, a portal with functionality for selecting content that was synchronized with the client. The concept said that this should be configured from the desktop computer and this is what achieved. The portal was shown for a couple of different companies and got positive feedback and this was the meaning. The client was unfortunately not finished enough to be shown together with the portal, this is a drawback since the interface between the client and server worked well. The project is however continuing and the client is getting more functionality. The work on the portal will also be continued after this project.

Documentation was not a requirement in the beginning and was written first when the development of the project had ended since a requirement of documentation was created. The reason for no documentation is to some parts due to there was no target that should read it; this is a common reason to that documentation not getting done [9]. Another reason is that documentation is not highly prioritized in agile development. Simple documents for specifications were delivered between the developers for the API’s.

Large parts of the concept’s ideas was missing, some due to that things took longer to implement than expected but many of them was not reasonable to achieve like buying media through the portal, this need a lot more investigations, negotiations and development work. Buying media through the portal is needed if the system should be sent to the market, partly because of that free content is limited but mostly to get third party providers integrated into the project to make it grow and generate cash flow. The amount of content on the portal was limited; the content that was added was there as placeholders for each of the category to make a quick overview of the system look nice and to test the different parts of the system. To attract customers and to see if design changes need to be done in the listing of content more content need to be added. The listing of content structure was a little evolutionary since the way content was distributed was not known when starting the design of the system.

The graphical interface had low priority in the project even with the fact that drag and drop was pointed out to be important. The concept version had a drag and drop interface; this did however not seem to fit into the layout of the new system. A place where it could have fit is the playlist section but since this part only was done to show that playlists can be created based on the file system of the client more time was not spent in this section to created extended functionalities. More graphical components is needed, when comparing content in the portal towards other systems as iTunes, media as music often has pictures of the music album cover or a short description of the content, time for adding such functionality or content were not planned.

Another user experience related functionality that was missing is the possibility to add own podcasts or news feeds that you have found on internet. This is something that would be needed to keep customers satisfied. The functionality for adding such things exists in the administrator interface and the database and backend have support for content that belongs to a certain users. The part that is missing is the adding of content from the customer interface, the reason for this is that the code for receiving such input needs to be fail proof. This to avoid users sending faked content or creating some kind of denial of service attacks by adding a lots of podcast links towards a single site that do not distribute podcasts. The denial of service attacks would then be based on that the portal systems podcast parser would contact that server to get the podcasts which does not exist. In iTunes this is done on the client so such attacks cannot be done. A solution on this could be to have people inspect and approve the content that users submit, this would mean a short delay but they would probably accept this. The video site http://www.youtube.com works like this when uploading video files; the video is not published at once when uploaded.

The last pieces of functionality that was added to the portal did not have any backend, for example the booking of car services. This was an interesting area that was there to broaden the portal from just being an iTunes copy. There was however never any concrete ideas how service bookings and paying of parking should be solved.

The portal is missing some quite fundamental functionalities, one is that the user cannot move the local music collection to the car through the portal. It can be done through a USB memory stick but transferring it through the portal would have been a more optimal solution. This is almost not possible since it should mean a lot of traffic. A solution could have been to move it over the local WLAN if it was available but anything in this direction was never designed. The portal can manage playlist of the files located on the client when they have been transferred through another medium. Another way to manage personal media is to have it on a public web server, no such services where investigated but global storages on the net are on the way. This conclusion can be drawn based on that the free email services http://www.gmail.com and http://www.hotmail.com have expanded their storages a lot the nearest time and that services for storing music might show up.

There were ideas for the system to handle information from third party applications both on the client and the server; this could for example be input from the car hardware as tire pressure. This information should then be submitted to the server and taken care of by a plug in or module. Coding on this was started but a concrete application example or a greater study would be needed to create something that would have worked.

6.5 Future work

The continuation of the portal is planned and the company is searching for more thesis workers than can continue to develop it. Two persons are already writing a thesis on how the security should be solved through communication between the client and the server. Security is an area that has not been taken much time in the design. The portal should be safe towards SQL-injection [32] and passwords are encrypted but secure connections are not used in the communication so the portal is not safe at the moment. This can be added as layer on top of the portal so it should not be a problem for future work. The authentication between the client and server is fairly simple and need to be reconstructed. The reason why security got a low priority is because of the system only will be shown in closed session and not going public.

More time should be spent on the user interface to make it intuitive, in this project the look and feel had low priority, some graphical elements was created to attract viewers and to make it easier for a user to overview the content selection. A company that is specialized on graphics has been contacted to create a new user interface for the portal.

The future of the portal is a little bit unclear, work will continue to improve it but to integrate more third party services it will probably need to be redesigned. A solid base has been created and the communication over RSS is something that can be reused in future implementations, it is a clear format and easy for the client to implement. A new server implementation should not have difficulties to replicate an own feed.

7. Conclusion

This chapter will draw conclusions of the outcomes of the project. The positive feedback from the demonstration of the portal shows that there exist an interest and demand for this kind of services in the automotive industry. The portal is however far from finished. Functionality for retrieving and selecting media exists but a lot more content is needed to make the service attractive on the market. To get a market segment much effort in retrieving content and structure it for easy access for the customers is needed. Services for buying content is also something that should be integrated to allow third party companies take part of the publishing service. No such external services was studied enough to be integrated in the portal since no open API's were found. And since this was only a proof of concept prototype this was left out.

To get an attractive service, third party content providers need to open their systems so they can be integrated. Sony Ericsson Play Now was investigated to be integrated in the system and the conclusion was that it could not be integrated as it looks now but with a tiny change the portal could probably make a bridge for adding media to the client. Another approach to get access to all the content provided through Play Now would be to build a webpage that worked directly in the client. The question then would be how should bought content be distributed to multiple cars all sharing the same account in the portal.

Continuing studies on how the portal should handle customers who own multiple cars and have many different drivers. Studies on what content the portal should handle need to be done; decisions need to be made for this to continue the project.

Early in the project the decision was taken to limit the project to only handle a client that constantly is online. A client that loses its connection to the internet and portal would need to buffer for example web radio stations to continue using them.

The future for a Wireless Infotainment System in a car is probably something that waits behind the door, mobile internet connections is getting cheaper and the connection coverage is increasing. Other products are integrating to the internet, phones are getting bigger displays and cameras are prepared to upload videos directly to internet, why should a technical advanced vehicle only offer the user replay of music on CD? How would it affect a person when taking a seat in the car gets the favorite comic on the screen, a short note of today's news and instead of the local radio stations can listen to the favorite radio station from internet!

To answer the initially asked question, what services that is suitable to integrate is all those with a working distribution channel. This project has shown that music, audio books, podcasts and RSS-feeds works fine to distribute freely. Streaming services as web radio and video will however need more technology for buffering to be deployed in a live environment.

References

[1] MacCormack, A, *Product-Development Practices That Work: How Internet Companies Build Software*, MIT Sloan Management Review; Winter 2001; 42, 2; pg.75, 2001

[2] *eMotor.se, Volvo öppnar en ingång för iPod-användare*, http://www.emotor.se/nyheter/visa.php?1388, [accessed 2008-11-24]

[3] *Apple – iPod + iTunes*, http://www.apple.com/itunes/, [accessed 2008-11-24]

[4] *Sony Ericsson – Pressmeddelanden – Översikt*, http://www.sonyericsson.com/cws/companyandpress/pressreleases/pressrelease/pressreleaseoverview/playnowarena-20080825?cc=se&lc=sv, [accessed 2008-11-24]

[5] Sommerville, I, *Software Engineering*, Addison-Wesley, 2007

[6] Maylor, H, *Project Management*, Prentice Hall, 2005

[7] *MP3.com – the source of digital music!*, http://www.mp3.com/feature/aboutus/, [accessed 2008-11-24]

[8] *RSS 2.0 Specification*, http://cyber.law.harvard.edu/rss/rss.html, [accessed 2008-11-26]

[9] AvNancy M. Dixon, *Common Knowledge: How Companies Thrive by Sharing what They Know*, Harvard Business Press, 2000

[10] *Cybercom Group*, http://www.cybercomgroup.se, [accessed 2008-12-05]

[11] *IPTC Web*, http://www.iptc.org/cms/site/index.html;jsessionid=axspRUp2ejP9?channel=CH0106 [accessed 2008-12-05]

[12] Lucas Gonze, *A survey of playlist formats*, http://gonze.com/playlists/playlist-format-survey.html, [accessed 2008-12-05]

[13] *P3 Christer*, http://www.sr.se/Podradio/xml/p3_christer.xml, [accessed 2008-12-05]

[14] *GP/Göteborg*, http://www.gp.se/rss/index.jsp?d=113, [accessed 2008-12-05]

[15] Bengtsson I., 2000. *Failures in telematic projects* [Conversation] (Personal communication, 2 December 2008)

[16] *Apache Cayenne,* http://cayenne.apache.org, [accessed 2008-12-06]

[17] *Apache Tapestry*, http://tapestry.apache.org, [accessed 2008-12-06]

[18] *Java Servlet Technology*, http://java.sun.com/products/servlet, [accessed 2008-12-06]

[19] *QT Cross-Platform Application Framework*, http://trolltech.com/products, [accessed 2008-12-07]

[20] *Arora*, http://code.google.com/p/arora, [accessed 2008-12-07]

[21] *PHP*, http://se.php.net, [accessed 2008-12-07]

[22] *Windows Embedded CE Overview*, http://www.microsoft.com/windowsembedded/en-us/products/windowsce/default.mspx, [accessed 2008-12-07]

[23] *BMW X5 : Catalogue*, http://www.bmw.com/com/en/newvehicles/x5/x5/2006/allfacts/catalogue.html, [accessed 2008-12-08]

[24] *XML Introduction*, http://www.w3schools.com/xml/xml_whatis.asp, [accessed 2008-12-09]

[25] *Apple Reports Forth Quarter Results*, http://www.apple.com/pr/library/2008/10/21results.html, [accessed 2008-12-09]

[26] *App Store Downloads Top 100 Million Worldwide*, http://www.apple.com/pr/library/2008/09/09appstore.html, [accessed 2008-12-09]

[27] *Viktoriainstitutet*, http://www.viktoria.se/files/Program_v2.pdf, [accessed 2008-12-09]

[28] *iTunes Store Tops Over Five Billion Songs Sold*, http://www.apple.com/pr/library/2008/06/19itunes.html, [accessed 2008-12-10]

[29] *OsCommerce, Open Source Online Shop E-Commerce Solutions*, http://www.oscommerce.com, [accessed 2008-12-11]

[30] *phpBB – Creating Communities Worldwide*, http://www.phpbb.com, [accessed 2008-12-11]

[31] *Subversion*, http://subversion.tigris.org, [accessed 2008-12-11]

[32] *SecuriTeam – SQL Injection Walkthrough*, http://www.securiteam.com/securityreviews/5DP0N1P76E.html, [accessed 2008-12-11]

Appendix A: Keywords

API	Application Programming Interface, used for specifying communication between systems.
Cron Job	Scheduled execution of a program in a UNIX environment.
Keep-alive	Connection that is open between client and server even that no data is sent.
Podcast	Audio files that often contains prerecorded radio programs.
Query String	Parameters for a dynamic webpage set after the '?', e.g. *.com?a=b*
Servlet	Java program deployed and run on a web server [18].
Shell Script	Tiny program run on a UNIX platform.
Tag	Used in for example HTML and XML to define structure, *<tag>* symbolizes start and *</tag>* the end of a element.
Telematic	Communication in automotive industry with for example vehicle data or position.
Web radio	Stream of music distributed on the net just as a radio channel.
Widget	Tiny application.
XML	EXtensible Markup Language, using tags to create data structures, the tags are not predefined. [24]

Appendix B: User Documentation

USER DOCUMENTATION

1. Introduction

This document describes how the web portal will be used, from a customer's view, a salesman and an administrator's view. Since the system is web based the user should type the address of the server the system is deployed on into the web browser to access it.

2. Customers perspective

To access the portal the user need to login to the system, this is done by filling in username and password in the form that is present. The user can check a checkbox called *Remember me* to save login information for the next visit.

When logged in the user is met by a page showing the latest content that has been added to the system. By clicking on a link next to one of these media contents the user will be taken to a page showing more information. For more information on selecting content, see the chapter 2.2.

To find more content there is a menu listing all the content to the left. By navigating in the menu sub categories is shown. When opening a sub category the content in that category is shown to the right.

If uncertain in which category content exists the interface allows the user to search for content. By typing a keyword in the form at top right of the page and clicking search the database will be searched for content including the text entered.

2.1 Configuration

To change configurations for the portal and the client the user should use the link called *Configuration*, then a new page is shown called Configuration is shown.

Settings
To change settings click the link titled *Settings*, then a form is shown were settings can be changed. To save the chosen settings click the *Save* button.

Themes
To change theme of the client click the link titled *Theme Configuration*, then the available themes are shown. The selected theme has a tinted background. To preview a theme, click the image next to the title of the theme. To select a theme click the link titled ✓ *Select this theme*.

2.2 Selecting, Previewing and Deselecting Content

When content can be selected by clicking the icon titled *Add to My Selections*. To preview the content the icon titled: *Preview* can be clicked; this takes the user to the third party provider's site. If a content item is selected and the user wants to deselect it the icon titled: *Remove from My Selections* should be clicked.

2.3 Playlists

It is possible to create playlists that later is accessible on the client from the interface. By clicking the link *Playlists* under the category *My Selections* in the menu to the left the Playlists page is shown.

To create a playlist the name is typed into the text box in the form to the upper right. Then the button *Create new playlist* is clicked.

To add tracks to a playlist the list first has to be selected in the table to the right. When a playlist is selected an *Add to playlist* link in the file listing to the left is shown. By clicking this link the file will be added to the list.

To remove tracks from a playlist the list first has to be selected in the table to the right. Then the current tracks in the list is listed in the table to the right with a link named *remove*, by clicking this link the item will be removed.

2.4 Logging out

To logout the link titled *Logout* in the menu at the top of the page should be clicked.

2.5 Icon Summary

✓	*Add to My Selections*
👁	*Preview*
🗑	*Remove from My Selections*
🔄	*Subscribe to this feed*
✗	*Stop your subscription to this feed*

3. Salesman Perspective

When logging in as a salesman a list of the users and profiles of the system is shown. A form for adding new users is shown below titled *Add new car*. By selecting a user selections for it can be done by getting information from a predefined profile. Three such profiles exists, Pre Family, Family, Post Family. By clicking one of them the content that has been selected for this profile is shown. To add this content for the selected user and profile, the link *Select Pre Family* should be selected. Named depending on which predefined profile that is selected.

3.1 Changing of preselected content

To change the content of a predefined profile logout and then login as this profile and then content can be selected as a normal user.

4. Administrators Perspective

When logged in as an administrator a administrator interface is available, this chapter describes the different possibilities in the interface. The chapters are named as the menu links are named in the portal administrator interface.

4.1 Content

4.1.1 Tree

In the content page the menu located at the left can be edited under the title *Content-tree*. A few icons are clickable and handle the editing of the tree.

✖	Delete the current item
▲	Move the current item one step up
▼	Move the current item one step down
▼✚	Add a tree item in this category
Edit	Edit the current item

To add content a category is selected in the tree, then two forms is shown to the right. If a podcast should be added the *Add podcast* form should be used, otherwise the *Add content* form should be used. When the form has been filled the *Submit* button should be clicked. This adds a content group, now to add a content item to this content group the content group needs to be located and selected in the tree. When the content group is selected a form is shown to the right called *Add content_data* this should now be filled.

4.1.2 Podcast

In the podcast page all available podcasts are shown, the listing shows when the podcast was fetched from the remote site, a status flag says if it failed or not. A zero (0) means that it was ok. The interface allows for updating a single podcast, clearing the items that this podcast have in the local database. It also allows the administrator to update all podcasts. Podcasts can be previewed by clicking the podcast title.

4.1.3 Categories

The categories page is static and only shows the id each category the database contains, an icon of the category and the title it has.

4.1.4 Enclosure Types
The enclosure type's page is static and only shows the id each enclosure type the database contains, an icon of the enclosure type, the title it has and its description.

4.1.5 Subscriptions
The subscription page is static and shows which subscriptions is currently active and their status.

4.1.6 Mp3.com
The mp3.com page gives links for parsing content on the mp3.com site, it also shows which items that is contained in the database.

4.2 Configuration

4.2.1 Settings
The settings page gives the possibility to create settings that the user can access from the Configuration-Settings page in the portal. The screen show the current settings to the left where they can be edited or removed.

In the middle a form for adding new settings is visible.

To the right a preview of how the user will see the settings is shown.

4.2.2 Themes
Functionality for handling themes was not implemented.

4.3 Logs
The log page shows a table of what error messages that has been reported by the system.

4.4 Users

4.4.1 List
In the user listing users and profiles can be edited, to edit a user the icon is clicked next to the first profile name of the user. All profiles for each user is listed.

Editing a user is done by clicking its icon in the listing. Then a form is shown for each profile the user has. Only one profile can be updated at the time. The changes done that is shared between the profiles (the user info, e.g. Registration no) is updated for all.

4.4.2 Add

To add a user the link titled *Add* should be clicked, when filling the form both a user and profile will be created.

4.4.3 User Classes

The following user classes are present:

Default User
Unique user used as a owner of all common content, should to be unique

Normal User
The normal user is connected to a car; multiple profiles can be bound to one user which all has different usernames and passwords. They profiles will share the selected content that need to be downloaded. Smaller entries as themes, news and web radios will be individual.

Administrator
The administrator's role is to add content; they can update podcasts and edit the menu tree. They can also edit users, create settings and check logs.

Salesman
To simulate an interface for a car salesman that uses the portal from the store a user of that class exists. It has the possibility to create cars (users) and add profiles to them. Then a profile with pre selected content can be copied to the created user.

Pre-defined
This user class is a container for selected content; they appear as a normal user in the system and can select content. Then a Salesman can copy the profile to a newly created user instead of selecting all content again. There are three default profiles; Pre-Family, Family, Post-Family.

4.5 Help

No help page has been implemented.

Appendix C: System Documentation

SYSTEM DOCUMENTATION

This document is pointed towards the one that should maintenance the system; this document includes specifications for the system and information about the environments.

1. Requirements

1.1 Server

The server portal requirements is that it should provide a end customer with an interface for selecting media that then is provided to a client.

2. System Specification

Three ways into the system should be created, one portal the customer accesses from a desktop computer, where the content can be selected and themes can be changed. It is named Web-portal in this document. One portal that is accessed from the clients simple interface, this should be kept simple to allow the customer to navigate the portal through a tiny touch screen. This portal is named WIP-portal in this document. An API should be created that the client connects to and received what content the customer have chosen. This is called Feed in this document.

Graphical user interfaces should be kept simple and have low priority but they should look good enough to attract investors.

Web-Portal

The Web-Portal is accessed both by customers, car salesman and administrators, they have different purposes in the system. The users should customize their media experience. Salesman should be able to do a initial configuration on a newly created user. Administrators should be able to change content in the system.

User Interface

Login functionality for multiple user classes
Menu tree for navigating through content
Search functionality for searching after content
Possibility to upload data from the client
Manage different types of media content
Change settings and theme
Creation of playlists

Salesman Interface

A salesman should be able to create new users and profiles.
By selecting predefined profiles, preselected content should be copied to a newly created user.

Administration Interface

Allow an administrator to add and edit content
Changing of the menu tree where content is stored
Adding and editing users
Check error logs
Create settings

WIP-Portal

A simplified portal that is accessible by a device using a small touch screen, created as a placeholder. The portal should give the customer possibility to select new content.

Feed

The feed should be able to distribute all the content that the customer has selected in the portal systems. The distribution is done by linking to the third party sources where the content is placed. The feed should mark content with the data when it was changed or added so the client knows what content to download or update.

The most important specification is the API used in communication between the client and server.

3. Detailed System Specification

3.1 Communication

3.1.1 RSS-feed

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type='text/xsl' href='feed.php'?>

<rss version="2.0">
  <channel>
    <title>WIP rss</title>
    <link>http://www.cybercomgroup.se</link>

    <description></description>
    <language>en</language>
    <copyright>Cybercom Group</copyright>
    <lastBuildDate>Wed, 12 Nov 2008 12:58:38 GMT</lastBuildDate>
    <category>WIP feed</category>
    <generator>WIP web version 0</generator>
```

Figure 1: The header of the main feed

Explanation of tags from Figure 1:

lastBuildDate	Date when last change to the feed was done

```
    <item>
      <title>RSS</title>
      <link><![CDATA[http://x.x.x.x/wip/v2/feed_rss.php]]></link>
      <description></description>
      <author>auth</author>
      <category>rssfeeds</category>
      <guid isPermaLink="false">rss12345</guid>
      <pubDate>Wed, 12 Nov 2008 12:58:38 GMT</pubDate>
      <comments></comments>
      <source></source>
    </item>

    <item>
      <title>Chapter 1-2</title>
     <link><![CDATA[x.x.x.x/audiobook/Tom_Sawyer_mp3_complete/Tom_Sawyer_01-02.mp3]]></link>
      <description>Tom Sawyer by Mark Twain</description>
      <author>auth</author>
      <category>audiobook</category>
      <guid isPermaLink="false">cd_6</guid>
      <pubDate>Tue, 09 Sep 2008 12:41:48 GMT</pubDate>
      <comments></comments>
      <source></source>
    </item>
```

Figure 2: Two examples of items contained in the feed, first a link to a sub feed for news and secondly an audio book stored locally.

Explanation of the tags of the items from Figure 2

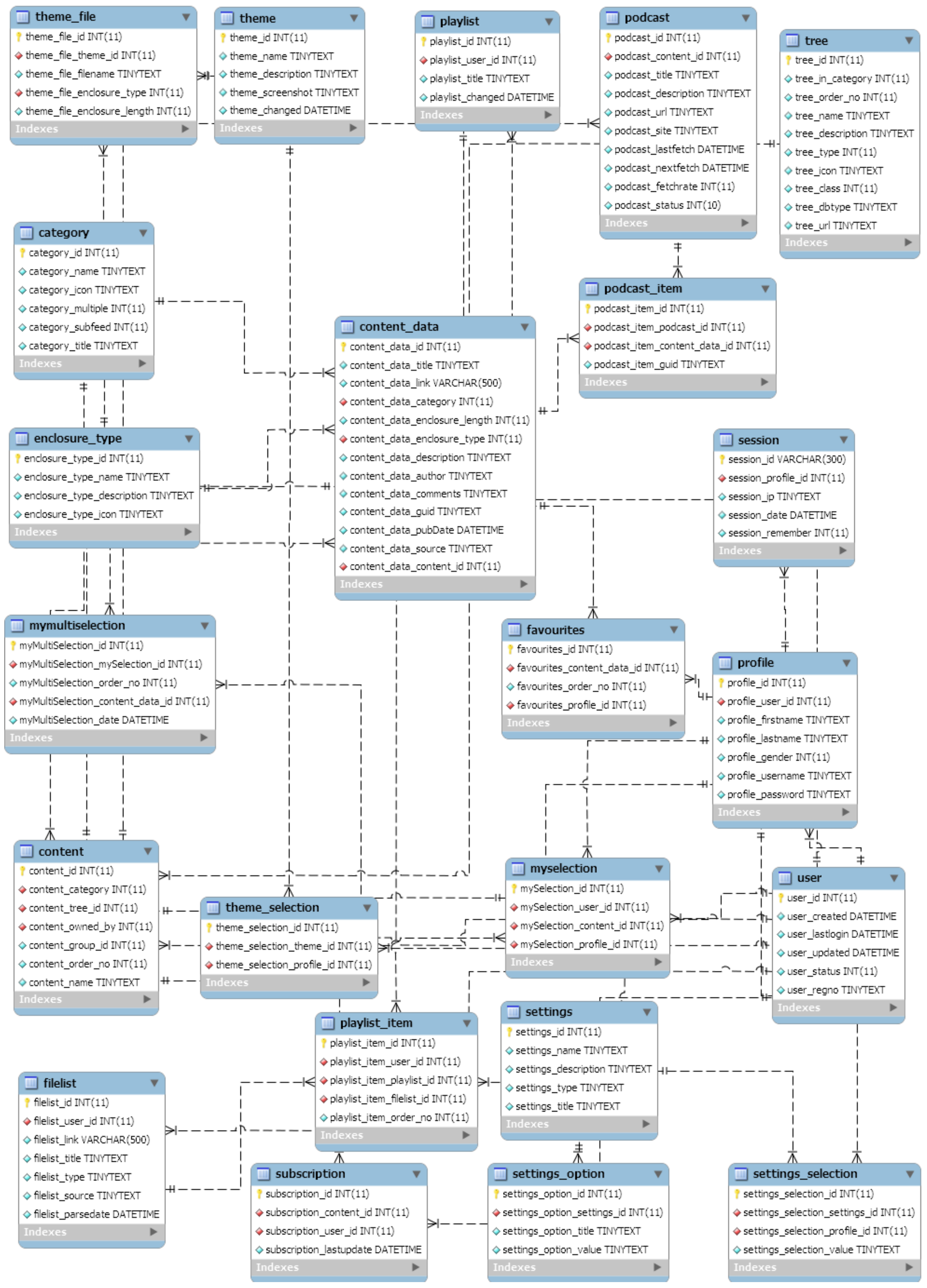
Title	Used as file name on client together with the link filename
Link	path to target
Description	used for choice of directory on the client
Author	not used
Category	the category the file should be sorted under
Guid	unique identifier for the item, based on table and its id
Pubdate	date of when the item was added to the database
Comments	not used
Source	not used

3.1.2 Data upload

Uploading of data is done by an http file upload; to identify files an extra textbox with the name *guid* can be used. The uploaded file should be identified by the name *userfile*.

3.2 Data Structures

3.2.1 Database



theme_file
theme_file_id INT(11)
theme_file_theme_id INT(11)
theme_file_filename TINYTEXT
theme_file_enclosure_type INT(11)
theme_file_enclosure_length INT(11)
Indexes
theme
theme_id INT(11)
theme_name TINYTEXT
theme_description TINYTEXT
theme_screenshot TINYTEXT
theme_changed DATETIME
Indexes
playlist
playlist_id INT(11)
playlist_user_id INT(11)
playlist_title TINYTEXT
playlist_changed DATETIME
Indexes
podcast
podcast_id INT(11)
podcast_content_id INT(11)
podcast_title TINYTEXT
podcast_description TINYTEXT
podcast_url TINYTEXT
podcast_site TINYTEXT
podcast_lastfetch DATETIME
podcast_nextfetch DATETIME
podcast_fetchrate INT(11)
podcast_status INT(10)
Indexes
tree
tree_id INT(11)
tree_in_category INT(11)
tree_order_no INT(11)
tree_name TINYTEXT
tree_description TINYTEXT
tree_type INT(11)
tree_icon TINYTEXT
tree_class INT(11)
tree_dbtype TINYTEXT
tree_url TINYTEXT
Indexes
category
category_id INT(11)
category_name TINYTEXT
category_icon TINYTEXT
category_multiple INT(11)
category_subfeed INT(11)
category_title TINYTEXT
Indexes
podcast_item
podcast_item_id INT(11)
podcast_item_podcast_id INT(11)
podcast_item_content_data_id INT(11)
podcast_item_guid TINYTEXT
Indexes
content_data
content_data_id INT(11)
content_data_title TINYTEXT
content_data_link VARCHAR(500)
content_data_category INT(11)
content_data_enclosure_length INT(11)
content_data_enclosure_type INT(11)
content_data_description TINYTEXT
content_data_author TINYTEXT
content_data_comments TINYTEXT
content_data_guid TINYTEXT
content_data_pubDate DATETIME
content_data_source TINYTEXT
content_data_content_id INT(11)
Indexes
enclosure_type
enclosure_type_id INT(11)
enclosure_type_name TINYTEXT
enclosure_type_description TINYTEXT
enclosure_type_icon TINYTEXT
Indexes
session
session_id VARCHAR(300)
session_profile_id INT(11)
session_ip TINYTEXT
session_date DATETIME
session_remember INT(11)
Indexes
mymultiselection
myMultiSelection_id INT(11)
myMultiSelection_mySelection_id INT(11)
myMultiSelection_order_no INT(11)
myMultiSelection_content_data_id INT(11)
myMultiSelection_date DATETIME
Indexes
favourites
favourites_id INT(11)
favourites_content_data_id INT(11)
favourites_order_no INT(11)
favourites_profile_id INT(11)
Indexes
profile
profile_id INT(11)
profile_user_id INT(11)
profile_firstname TINYTEXT
profile_lastname TINYTEXT
profile_gender INT(11)
profile_username TINYTEXT
profile_password TINYTEXT
Indexes
content
content_id INT(11)
content_category INT(11)
content_tree_id INT(11)
content_owned_by INT(11)
content_group_id INT(11)
content_order_no INT(11)
content_name TINYTEXT
Indexes
myselection
mySelection_id INT(11)
mySelection_user_id INT(11)
mySelection_content_id INT(11)
mySelection_profile_id INT(11)
Indexes
theme_selection
theme_selection_id INT(11)
theme_selection_theme_id INT(11)
theme_selection_profile_id INT(11)
Indexes
user
user_id INT(11)
user_created DATETIME
user_lastlogin DATETIME
user_updated DATETIME
user_status INT(11)
user_regno TINYTEXT
Indexes
playlist_item
playlist_item_id INT(11)
playlist_item_user_id INT(11)
playlist_item_playlist_id INT(11)
playlist_item_filelist_id INT(11)
playlist_item_order_no INT(11)
Indexes
settings
settings_id INT(11)
settings_name TINYTEXT
settings_description TINYTEXT
settings_type TINYTEXT
settings_title TINYTEXT
Indexes
filelist
filelist_id INT(11)
filelist_user_id INT(11)
filelist_link VARCHAR(500)
filelist_title TINYTEXT
filelist_type TINYTEXT
filelist_source TINYTEXT
filelist_parsedate DATETIME
Indexes
subscription
subscription_id INT(11)
subscription_content_id INT(11)
subscription_user_id INT(11)
subscription_lastupdate DATETIME
Indexes
settings_option
settings_option_id INT(11)
settings_option_settings_id INT(11)
settings_option_title TINYTEXT
settings_option_value TINYTEXT
Indexes
settings_selection
settings_selection_id INT(11)
settings_selection_settings_id INT(11)
settings_selection_profile_id INT(11)
settings_selection_value TINYTEXT
Indexes

Figure 3: Diagram over the database, arrows shows how the tables are linked

Below follows a short description of the database tables shown in Figure 3:

Category
Used for specifying which category content belongs to. *Subfeed* tells if it should be put in a sub feed when distributed.

Content
Groups of content

Content_data
Content items, grouped by content

Enclosure_type
Used to specify icon that should be shown in the interface and which type that should be specified in the feed.

Favourites
Used for highlighting certain *content_data* items

Filelist
Handles files present on the client for creation of playlists

Mymultiselection
Selection of items in the *content_data* table

Myselection
Selection of content groups in the *content* table

Playlist
Used for naming and identifying playlists

playlist_item
Items grouped by *playlist* table.

Podcast
Information about a created podcast, used by the podcast parser

podcast_item
Item linked to *podcast* and *content_data* used for possibility to remove podcast items from the *content_data* table and to make sure duplicates not are retrieved when parsing a remote podcast

Profile
Handles usernames and passwords

Session

Used for users connected to the portal and for remembering users between sessions

Settings
A settings item

Settings_option
Option for a settings element, multiple options can be created for one settings item

Settings_selection
User selection of an *settings_option*

Subscription
Handles user subscriptions of podcasts

Theme
Identifies theme

Theme_file
File that belongs to a theme

Theme_selection
Handles user selection of themes

Tree
The menu tree

User
User table, which represent cars in the system

3.3 Code

The system is built in PHP code, a plain functional format is used divided into subdirectories.

Code style

Variables are normally lowercase, spaces ‘ ‘ are replaced with underlines ‘_’.

```
        $user;
        $user_id;
```

Global variables should start with an underline, global constants should be uppercase.

```
        $_LOGINPAGE;
```

File layout

Layout files should be separated from other functions. Files accessing the database with queries should not contain any other code.

4. Development Environment

As development environment Eclipse PDP Project has been used, it can be downloaded from http://www.eclipse.org/pdt/. An installation guide can be found on the site.

When installed a new project should be created, then the source files can be imported.

5. Server Environment

As operation system the GNU/Linux distribution Centos 5.2 has been used.
http://www.centos.org/

Apache HTTP Server Project has been used as web server.
http://httpd.apache.org/

MySQL has been used as database engine.
http://www.mysql.com/

PHP has been used for scripting language.
http://www.php.net/

The server runs on an ordinary PC.

5.1 Installation

Here follows a few hints of what to select when going through the installation of the operating system. There appeared some errors when installing the systems, so if these occur again the solution for them is listed in the text below.

When the installation executes a few options is shown, choose these two

```
server
mysql
```

When the installation is finished other software should be installed and configured.

5.1.2 Startup commands

Commands for starting services that should be used is listed below

```
httpd
service mysqld start
```

5.1.3 Apache

The configuration file for apache can be found in

```
    /etc/httpd/conf/httpd.conf
```

The root dir for public web pages is found in

```
    /var/www/html
```

5.1.4 MySQL

Error:

host not allowed...

Solution:

```
    GRANT ALL PRIVILEGES ON *.* TO 'root'@'hostname'
    IDENTIFIED BY 'admin';
```

Create user

wip_v1:<password>

If moving the tables from one database to another, tables tend to change names to lowercase, this have to be fixed manually by these two sql queries:

```
    ALTER TABLE `wipv2`.`mymultiselection` RENAME TO
    `wipv2`.`myMultiSelection`;

    ALTER TABLE `wipv2`.`myselection` RENAME TO
    `wipv2`.`mySelection`;
```

5.1.5 PHP

Error:

mysql_connect – undefined

Solution:

```
    yum -y install php-mysql.i386
```

Error:

dom document failed (podcast_parser.php)

Solution

```
    yum -y install php-xml.i386
```

Error:

phpmyadmin complains over mbstring

Solution

```
    yum -y install php-mbstring.i386
```

After installations of the new packages the web server need to be restarted
```
httpd -k restart
```

5.2 Useful Commands

Below follows a few commands that can be useful in the server environment.

To list packages that are available for installation this command is used:
```
yum list
```

Exporting of data locally
```
mysqldump --default-character-set=latin1 wipv2 -u
wip_v1 -p > sql.sql
```

Importing of data on the server
```
mysql -u wip_v1 -p wipv2 < sql.sql
```

Change network settings
```
system-config-network
```

6. Functionality still to implement

Most of the planned functionality on the server is implemented

Here follows a list of tasks that never were crossed of the priority-list.

- New expanding listing interface for My Selections
- Theme of the portal following client theme
- Wizard for pre selections
- Improved listing of directories
- Feed for setting files that the client should download and upload
- Scheduling for podcast fetching
- Expanded search functionality as saving of searches
- Improved desktop

These features can be seen as minor, some more functionality on the files sharing between the client and server is needed to handle third party applications that should handle data that is send from the client. For example functions like diagnostics of the client.

6.1 Known Bugs

The podcast parser codes with wrong character coding.
Probably a simple fix by just removing the character encoding that was used when the system was deployed on windows.

Autologin to the WIP-portal, user is forwarded to web portal

Needs testing with the client to find problem

Popup window on WIP-portal is not shown when scrolling
The WIP-browser is interpreting javascript different than arora and firefox, need to be tested on the client to find problem.