NTNU
Innovation and Creativity

Interactive Television on Handheld Devices
Handling of metadata and creating interactivity in T-DMB and DVB-H

Andreas Engen Andersen

Master of Science in Electronics

Submission date:	September 2007
Supervisor:	Andrew Perkis, IET
Co-supervisor:	Peder Drege, Adactus

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

DVB-H [21] and T-DMB [24] are two standards that both form the basis for enabling broadcast of video content to handheld devices. In order to provide the user of these services with more than just the video content, the possibility to deliver an interactive presentation and communicate with the user is important.

The student shall identify the various components in the content value chain, investigate the strengths and weaknesses of each standard for true-time two-way communication, and discuss how the different standards provide the tools and protocols to achieve an interactive way of presenting content and metadata. The two standards shall be investigated both from a content providers'- and from an end-users' point of view, where especially the possibilities of a content provider to create his own services and deliver them to the end-user shall be investigated. A discussion shall also be made regarding what terminals/client software that according to the standards will be able to make use of this information automatically, and what the benefits would be for a content provider to create his own client applications and simply use the two standards to transport his content and metadata.

An implementation that shows how such metadata can create an interactive experience for the end-user shall be done using the tools defined in XML [45]. The metadata shall describe a scenario where interactivity is used for communication between an end user and a content provider, and where this metadata gives the end user a better service and/or a more relevant presentation. Metadata structure can be wrapped in other formats like MPEG-7 and/or MPEG-21 for delivery. The metadata that will be implemented will form the basis for the discussion of how DVB-H and T-DMB can be used for creating interactive services.

Assignment given: 07. May 2007
Supervisor: Andrew Perkis, IET

Abstract

As broadcasted television is changing from analogue to digital transmission, the television industry has to adapt itself for a new reality. Digitization opens for a wide array of new ways of watching television, where interactivity and mobility are paramount. The difference in the experience lies in the interactive part, inviting the user to take part in what happens on the mobile screen. What the mobile telephone lacks in screen size it can now make up for with its interactive potential. Instead of just watching television, the user now interacts with it. As a result, the television experience can be tailored to suit consumers with different requirements.

In this study I look at true-time broadcasted television to handheld devices over the standards achieved in Europe and Korea today, DVB-H and T-DMB, and how interactivity between content provider and end-user can be achieved. I also look into how metadata plays a crucial role in interactive television, and the means to utilize metadata to favor the end-user's demands according to standards such as XML, MPEG and Tv-Anytime. By supplying metadata to i.e sport or reality shows, and hence creating interactivity between content provider and end-user, a new marked for television is made possible. Electronic program guides (EPGs), teletext and weather forecasts for handhelds are examples of ways that metadata can be utilized to create an interactive experience for the end-user.

Preface

This Masters Thesis is written for the course "TTT4705 Multimediasignalbehandling Fordypning" at the Norwegian University of Science and Technology. The course is a part of the fifth and last year of a Master in Technology degree in the field of Telecommunications Engineering.

The purpose of this thesis was to study and implement interactive television on hand-held devices. The project lasted from mid-May to mid-September 2007.

Thanks to my primary supervisor, Professor Andrew Perkis for his motivation and support on the topic, and to CEO Peder Drege and CTO Thoms Skjølberg at Adactus for excellent guidance and help throughout the course of the project. I would also like to thank my friend Jonas Ask for helping me with thoughts, ideas and debugging of my software.

Trondheim, September 25, 2007

Andreas E. Andersen

Abbreviations

For the purposes of the present document, the following abbreviations apply:

3G	Third Generation wireless
API	Application Programmer Interface
BIFS	Binary Format For Scenes
Codec	Coder or decoder
DAB	Digital Audio Broadcasting
C/N	Carrier-to-Noise ratio
CLDC	Connected Limited Device Configuration
COFDM	Coded Orthogonal Frequency Division Multiplexing
CPU	Central Processing Unit
DigiTAG	Digital Terrestrial Television Action Group
DI	Digital Item
DID	Digital Item Declaration
DIDL	Digital Item Declaration Language
DOM	Document Object Model
DMB	Digital Multimedia Broadcasting
DTV	Digital Television
DTD	Document Type Definition
DVB	Digital Video Broadcasting Consortium
DVB-T	Digital Video Broadcasting - Terrestrial
DVB-H	Digital Video Broadcasting - Handheld
DVR	Digital Video Recorder
EDGE	Enhanced Data Rates for GSM Evolution
EPG	Electronic Program Guide
ESG	Electronic Service Guide
FEC	Forward Error Correction
MPE-FEC	Multiprotocol Encapsulation Forward Error Correction
RS-FEC	ReedSolomon Forward Error Correction

FLUTE	File Delivery over Unidirectional Transport
GPRS	General Packet Radio Services
GSM	Global System for Mobile communication
HTTP	HyperText Transfer Protocol
IOD	Initial Object Descriptor
IPDC	Internet Protocol Datacasting
J2ME	Java 2 Micro Edition
JAXP	Java API for XML Processing
JVM	Java Virtual Machine
JRE	Java Runtime Environment
MBMS	Multimedia Broadcast Multicast Service
MDF	Metadata Delivery Framework
MOT	Multimedia Object Transfer
MPEG	Moving Picture Expert Group
OD	Object Descriptor
OFDM	Orthogonal Frequency Division Multiplexing
PDA	Personal Digital Assistant
PIM	Personal Informational Management
PVR	Personal Video Recorder
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
SAX	Simple API for XML
SFN	Single Frequency Network
SGML	Standard Generalized Markup Language
SMIL	Synchronized Multimedia Integration Language
SNR	Signal-to-Noise Ratio
TCP	Transmission Control Protocol
TDC	Transparent Data Channel
T-DMB	Digital Multimedia Broadcasting - Terrestrial
TS	Transport Stream
UMTS	Universal Mobile Telecommunications Service
URI	Unified Resource Identifier
XML	eXtensible Markup Language
WAP	Wireless Access Control
W3C	World Wide Web Consortium

Contents

List of Tables

List of Figures

Part I

Introduction

Chapter 1

Introduction

The multimedia industry has experienced an enormous growth in the last few years. The reason for this is the huge increase in technologies which enables wireless communication equipment such as mobile phones to become a tool for everyday use. The mobile telephone today is a device made to fit most purposes in everyday life, replacing music players, calendars, calculators and similar equipment. All these technologies are now merged into one highly sophisticated Personal Digital Assistant (PDA) which you can make calls from, browse the Internet and even watch TV on. Instead of having a multitude of apparatuses organizing your everyday life, one now does it all.

This thesis presents an in depth study of interactivity and how it can be achieved on mobile phones by using predefined standards such as XML [45], MPEG-21 [1] and Tv-Anytime [42]. The report describes the history and development of these new and enabling technologies for interactive handheld television, shows how interactivity can create a whole new way of watching television *on the go*, and most importantly, outlines a system and feasible architecture for implementation of an interactive Electronic Program Guide (EPG) on a mobile phone.

1.1 Motivation

In *Mobile TV, the next european mega trend* [39], Eurescom mess@ge editor Peter Stollenmayer writes:

> "Thirty years ago, most of us had not considered it possible that everyone can talk to everybody from everywhere. Meanwhile this dream has become true for nearly a quarter of the world population who own mobile phones. Today, most of us cannot imagine that we could watch TV wherever we are. Tomorrow, this will be just as normal as making a mobile phone call."

The goal of this project has been to take a scientific approach to the two standards that form the basis for broadcast of video content to handheld devices in Europe today, and show how to create and implement interactivity between the content provider and the end-user through predefined standards such as XML [45], MPEG-21 [1] and Tv-Anytime [42].

Television has gone from analogue to digital broadcast, and that has enabled possibilities such as receiving content on handheld devices as well as on the television at home. However, functionalities for creating a more interactive enviroment between the content provider and the end-user has not been very well developed. This study is focusing on the possibilities to create custom-taylored interactive television content based upon what each end-user prefers to watch. This means that the user can watch what he or she wants, whenever he or she wants to.

With digital television the possibilities are endless, and with Electronic Program Guides (EPGs), weather forecasts and news highlights for handheld devices, the end-user can stay updated on his favorite television shows on his or her way to work.

1.2 Problem definition

"*DVB-H [21] and T-DMB [24] are two standards that both form the basis for enabling broadcast of video content to handheld devices. In order to provide the user of these services with more than just the video content, the possibility to deliver an interactive presentation and communicate with the user is important.*

The student shall identify the various components in the content value chain, investigate the strengths and weaknesses of each standard for true-time two-way communication, and discuss how the different standards provide the tools and protocols to achieve an interactive way of presenting content and metadata. The two standards shall be investigated both from a content providers'- and from an end-users' point of view, where especially the possibilities of a content provider to create his own services and deliver them to the end-user shall be investigated. A discussion shall also be made regarding what terminals/client software that according to the standards will be able to make use of this information automatically, and what the benefits would be for a content provider to create his own client applications and simply use the two standards to transport his content and metadata.

An implementation that shows how such metadata can create an interactive experience for the end-user shall be done using the tools defined in XML [45]. The metadata shall describe a scenario where interactivity is used for communication between an end user and a content provider, and where this metadata gives the end user a better service and/or a more relevant presentation. Metadata structure can be wrapped in other formats like MPEG-7 and/or MPEG-21 for delivery. The metadata that will be implemented will form the basis for the discussion of how DVB-H and T-DMB can be used for creating interactive services."

1.3 Report outline

This report is divided into three major sections, each of which will be presented below, along with a brief description of its contents. The main part of the unique work that has been performed in this project is described in the third part, *Own Contribution*, whilst the two other parts of the report deals with the *research* behind the standards used and the background for the work, and the *evaluation and conclusion* drawn from the study and implementation in this thesis.

1.3.1 Research

The research section describes the different standards that collaborate to make interactive television on handheld devices possible. It gives a thorough description of each of the two standards used for broadcast of video content to handheld devices in Europe, both DVB-H [21] and T-DMB [24], and where they stand today. The research gives an introduction to how they have grown from being technology show-offs at telecom trade shows in the year 2000, to becoming everyday use maybe as early as in 2008. Now the question is not *if* people will enjoy watching tv on their mobile phones, but what kind of shows that will be best suited for tv on mobile phones.

In my research I also look briefly into the MPEG standards [19] used in this thesis. When it comes to interactivity and description of metadata, I will look into XML [45], XML Schema [26] and Tv-Anytime [42].

The part *Creating Interactivity* describes ways to create interactivity on handheld devices such as mobile phones. An Electronic Program Guide (EPG) has been chosen as the main focus of this thesis, and how to keep the end-user up to date on what is aired at what time on his or her favorite TV channels.

1.3.2 Own Contribution

In the own contribution part of the report the reader will find a comprehensive analysis of how to provide interactive services on handheld devices, and how the end-user can choose to attend and be a part of something that is broadcasted on his or her handheld device.

This scenario will be followed by an analysis of where we are today, and what can be done to increase the interactivity between the end-user and the content provider by using XML [45] as a way to present metadata on handheld devices. I will also look into how interactive handheld television fits into the profile of Universal Multimedia Access (UMA) [28], a concept that aims at enabling access to multimedia content from any device through any accessible network. To maintain the demand of multimedia in a CLDC environment [14], the delivery of appropriate resources relies heavily upon the content provider.

An Electronic Program Guide made for handheld devices in a CLDC environment has been implemented in this thesis. The EPG was implemented using the J2ME programming language [22] and Eclipse framework [4]. The EPG shows how interactivity can be achieved on a mobile phone by utilizing metadata formatted in accordance with the XML standard.

1.3.3 Evaluation and Conclusion

In this section of the report I deal with the validity of the collected data and possibilities for further work along the lines drawn in this report. I also look into recent events where interactivity has become a crucial part of the mobile environment today.

Part II

Research

Chapter 2

Theory

In this section I will describe what *interactive television on handheld devices* implies to the end-user. I will look briefly into essential parts for enabling interactivity on handheld devices, such as XML [45], Tv-Anytime [42] and MPEG-21 [1]. I will look into what would be needed for end-users to receive interactive broadcasted television on their way traveling from home to work, and would like to watch tv on the buss or train.

I will also show how interactivity in the form of an Electronic Program Guide (EPG) can create a whole new way of watching television *on the go*. The concepts described in this chapter are quite broad, since covering every aspect would be far out of the scope of this thesis. Readers are referred to the bibliography and appendixes for a more thorough information on these concepts.

2.1 XML

XML [45] is short for eXtensive Markup Language, and is created by The World Wide Web Consortium (W3C) [25]. XML is a widely used markup language, and supports a wide variety of applications. XML is a simplified subset of Standard Generalized Markup Language (SGML), and its primary purpose is to facilitate sharing of data across different information systems. This is one of the reasons why it is so popular in multimedia context, since it is easy to structure data or descriptions of data (metadata) in elements by using tags. XML can also be used to create other markup languages, such as Synchronized Multimedia Integration Language (SMIL).

XML is as mentioned above made up of tags, which again defines various elements. The tags make up the markup, while the information is contained within these tags. This way it is easier to distinguish the markup from the information part. Another reason why it is so popular is that it is extensive, meaning there is no end to what you can do with it as long as you follow the rules.

XML is a markup language based upon strict syntactical rules, which must be followed in order for computer software to read (or parse) and understand the relative arrangement of information within the tags. Documents that follow these rules are described as *well-formed documents*, and must also have a definition of what structure that will be used and what they can contain. A definition like this is called a Document Type Definition (DTD).

2.1.1 XML Namespaces

When combining XML documents from more than one XML vocabulary, XML namespaces are used. An XML Namespace is a W3C [25] recommendation for providing uniquely named elements and attributes in an XML instance. If each vocabulary is given a namespace then the ambiguity between identically named elements or attributes can be resolved. To ensure unique element names, the different namespaces are identified by Unified Resource Identifier (URI) references used as prefixes. This way you don't need to include the URI in every tag of the element member. All element names within a namespace must be unique.

<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">

Table 2.1: Example of an XML Namespace

Table 2.1.1 shows how a namespace is used for the element DIDL. The namespace in the example is taken from *DIDL_soccer.xml* in appendix C, and is identified by the URI `"urn:mpeg:mpeg21:2002:01-DIDL-NS"`. The prefix *DIDL* can be used by all successors of the DIDL element that are from the same namespace, without the URI reference.

2.1.2 XML Schema

A way to apply additional rules to XML documents, is by using XML Schema [26]. XML Schema is written in XML and defines XML vocabulary, which in short means a set of rules to which an XML document must conform in order to be considered valid according to that schema. XML that follow XML Schema is called valid XML, and to validate a document against an XML Schema, a reference to the XML Schema must be included.

XML Schemas are more effective and precise compared to DTDs because of their capabilities of constructing stricter rules. The Document Type Definition (DTD) is a less complex method to define the strucure of XML Documents.

2.1.3 Document Object Model

The Document Object Model (DOM) is a platform- and language-independent standard object model for representing HTML or XML and related formats. The document and its elements are represented as nodes in a hierarchical tree-structure, and it supports navigation in any direction (e.g parent and previous sibling). DOM provides an interface for a programmer to the XML document through an Application Programmer Interface (API). The API contains methods to access, manipulate, load and save XML documents.

The DOM specifications are divided into levels, each of which contains required and optional modules. To claim to support a level, an application must implement all the requirements of the claimed level and the levels below it.

Because the DOM supports navigation in any direction as mentioned above and allows for arbitrary modifications, an implementation must at least buffer the document that has been read so far (or some parsed form of it). Therefore the DOM consumes a rather large amount of memory, and is likely to be best suited for applications where the document must be accessed repeatedly or out of sequence order. This means that DOM is typically used on smaller XML documents. If the application is strictly sequential and one-pass, the SAX model is likely to be faster and use less memory. SAX is an alternative programming interface for XML and is short for *Simple API for XML*. SAX provides more efficient analysis of large documents than DOM since it does not load the entire document into memory. Another advantage with SAX is that it is event-based instead of tree-based such as the DOM.

2.1.4 Parser

One of the most important parts of an XML application is the XML parser. The XML parser translates the document into useful data for the software to understand. The parser also validates the structure of the instance document against the DTD or XML Schema.

The parser structures the document and allows the user to control what parts it should focus on. In this project I will use XML Schema, and hence the namespace must be included for the parser to know what XML Schema to use. There are several available parsers, but in this thesis the kXML parser [12] was used.

2.2 Java 2 Micro Edition

The Java 2 Platform, Micro Edition (J2ME) [22] is a specification of a subset of the Java platform. J2ME is a product of Sun Microsystems and is aimed at providing a certified collection of Java APIs for the development of software for small, pocket-sized resource-constrained devices such as cell phones, calculators, PDAs and set-top boxes.

The combination of WAP (Wireless Application Protocol) and Java technology gives an ideal solution for interactive solutions on handheld devices, and gives the users the possibility to download third party applications to standard mobile phones. Sun provides a reference implementation of the specification, but has tended not to provide free binary implementations of its J2ME runtime environment for mobile devices, rather relying on third parties to provide their own.

2.3 Connected Limited Device Configuration

The Connected Limited Device Configuration (CLDC) [14] contains a strict subset of the Java class libraries, and is the minimal solution needed for a Java Virtual Machine (JVM) [16] to operate. CLDC is basically used to classify myriad devices into a fixed configuration. CDLC is described as:

> "A configuration provides the most basic set of libraries and virtual-machine features that must be present in each implementation of a J2ME environment.

When coupled with one or more profiles, the CLDC gives developers a solid Java platform for creating applications for consumer and embedded devices"

2.4 MPEG standards

MPEG (Moving Picture Expert Group) has since 1988 developed standards for multimedia, and the official designation of MPEG is ISO/IEC JTC1/SC29 WG11, but in this report I refer to the standard as MPEG.

MPEG-1 (1993), MPEG-2 (1996) and MPEG-4 (1999) are coding standards that have been created with production, consumption and distribution of multimedia content in mind. MPEG-7 and MPEG-21 however, are not. MPEG-7 is a description standard for multimedia content, and describes how to ease content management by adding metadata to the resources. MPEG-21 is the latest standard, and is still not finished. It provides a larger, architectural framework for creation and delivery of multimedia, and has seven key elements:

- Digital item declaration
- Digital item identification and declaration
- Content handling and usage
- Intellectual property management and protection
- Terminals and networks
- Content representation
- Event reporting

2.4.1 MPEG-21

There are some basic terms defined in the MPEG-21 framework [1] that we need to look closer into:

- Every transaction and all communication that exist are performed using *Digital Items* (DIs). The Digital Item is a structured digital object with a standard representation, identification and metadata within the MPEG-21 framework. This entity is also the fundamental unit of distribution and transaction within this framework.
- Every participant in the multimedia consumer and delivery chain is referred to as a *User*. A User interacts with Digital Items. According to *The MPEG-21 Book* [1], a User is:

  "Any entity that interacts in the MPEG-21 enviroment or makes use of a Digital Item."

- Multimedia assets such as image, text or movie clip is referred to as a *resource*. A Digital Item may contain several resources of different types. A resource is normally defined in a resource element by reference, by spesifying the Uniform Resource Identifier. URI identifies the resource so the program can find the content of the resource when called for.

In this thesis I am going to focus on part two of the MPEG-21 standard, Digital Item Declaration, and how MPEG-21 together with XML can form the foundation for interactive television over DVB-H and T-DMB.

2.4.2 Digital Item Declaration

According to the *The MPEG-21 Book* [1], a Digital Item (DI) is defined as a *structured digital object with a standard representation, identification and metadata within the MPEG-21 framework".* In the MPEG-21 framework, every transaction consists of Digital Items. When we look at MPEG-21 part two, it describes a model to describe the Digital Item ISO/IEC 21000-2, the Digital Item Declaration (DID). The Digital Item Declaration (DID) consists of three parts:

- The *DID Abstract Model*, which represents the general model to describe Digital Items. This model defines a set of abstract elements and concepts that form a useful and flexible model for declaring Digital Items.
- The *DID Representation of the model*, which consists of normative description of the syntax and semantics of every DID element in XML. In short, the Abstract Model is now implemented.
- The *Corresponding MPEG-21 DIDL Schemas*, which are well-formed, informative XML Schemas consisting of the entire grammar of the DID model and Digital Item Declaration Language (DIDL).

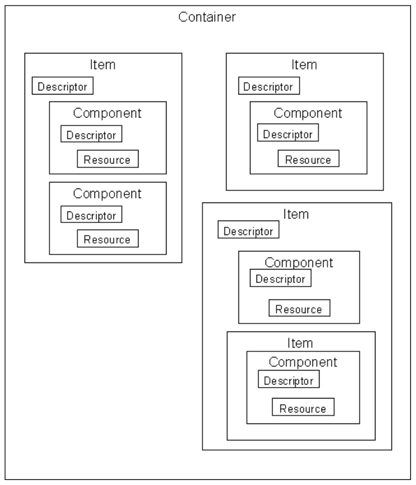


Figure 2.1: Example of an Abstract Model

In figure 2.1, we can see an example of how a Digital Item can be constructed according to the Abstract Model. The figure shows how the Items form a Container. Within this Container, an Item can contain different Components and a Descriptor which holds metadata about the different Components. Each of the Components again contains a Resource, which declares a resource either by reference or value.

Digital Items can be configured by adding choices, and thus being able to choose between several recourses that a Digital Item might contain. A choice is represented by Choice-Selection in the element structure. Several choices might be needed to select the very resource needed, allowing distribution of the same Digital Item to different terminals. A choice in a Digital Item representing a live soccer game on a mobile phone could be to choose information about the soccer teams, players, or voting for *Man of the Match*.

2.5 Tv-Anytime

A short description of the TV-Anytime Forum has been taken from the TV-Anytime Forum [42]:

> The global TV-Anytime Forum is an association of organizations which seeks to develop specifications to enable audio-visual and other services based on mass-market high volume digital storage in consumer platforms - simply referred to as local storage.
>
> The TV-Anytime Forum was formed at an inaugural meeting held in Newport Beach, California, USA, on 27-29 September 1999. It has started work to develop open specifications designed to allow Consumer Electronics Manufacturers, Content Creators, Telcos, Broadcasters and Service Providers to exploit local storage.
>
> As part of its formation, the TV-Anytime Forum has established four fundamental objectives for the organization, which are:
>
> - The TV-Anytime Forum will define specifications that will enable applications to exploit local persistent storage in consumer electronics platforms.
> - The TV-Anytime Forum is network independent with regard to the means for content delivery to consumer electronics equipment, including various delivery mechanisms (e.g. ATSC, DVB, DBS and others) and the Internet and enhanced TV.
> - The TV-Anytime Forum will develop specifications for interoperable and integrated systems, from content creators/providers, through service providers, to the consumers.
> - The TV-Anytime Forum will specify the necessary security structures to protect the interests of all parties involved.

2.6 Creating Interactivity

The focus in this thesis is upon interactivity, and how interactive television on handheld devices such as mobile phones can create a new way of experiencing television. An Electronic Program Guide (EPG) is the electronic equivalent of a printed television program guide. It shows an on-screen guide to scheduled broadcast television programs, and allows a viewer to navigate, select and discover content by time, title, channel or genre. All of this is done by use of the user's remote control, a keyboard or even a phone keypad as it will be shown in this thesis. The technology behind an EPG is based upon broadcasting metadata to an application within a set-top box, usually a software application made explicitly to handle metadata, which again connects to the television set and enables the application to be displayed. The EPG technology is predominant in the digital television and radio world.

Typical elements of an EPG comprise of a Graphical User Interface (GUI) which enable the display of program titles, descriptive information about each show (usually a synopsis), the channel name and program start and end times. The information is displayed on a grid with only the title showing, and with the option to select more information about each program. When EPGs are connected to Personal Video Recorders (PVRs) they enable a viewer to plan his or her viewing, which also gives the possibility to record broadcasted programs to a hard disk for later viewing.

EPGs metadata is typically sent within the broadcast transport stream or alongside it in a special data channel, and it contains the program start time and title as well as additional descriptive metadata. Newer media centers (PC based multi-channel TV recorders) and Digital Video Recorders (DVRs) may use an Internet feed for the EPG. This enables two-way interactivity for the user so that media download can be requested via the EPG, and remote programming of the media center can be achieved. An example is MythTV [35] and Elgato EyeTV [5].

The latest type of EPGs is a personalised EPG which uses semantics to be able to advise one or multiple viewers what to watch based on their interests. iFanzy [8] is one of those, and allows users to use and create custom skins (like a personal computer's desktop image). iFanzy is customizable and programmable, and can be operated to record all of the shows within a user's favorite categories, which means that the viewer no longer has to depend upon a broadcaster's time schedule.

Chapter 3

DVB-H vs T-DMB

"With the ongoing digitization of television systems based on the Digital Video Broadcasting standards, television evolves from pure audio-visual, time-linear services into rich media, interactive and time-shifted infotainment services. Due to the digital nature of the systems, transmitting arbitrary content, storing and accessing it becomes possible with any kind of digital multimedia-capable device."

This was written by *Uwe Rauschenbach* [34], and describes the essence of this report. I will in this chapter try to shed some light upon the two competing standards today.

3.1 Technical data about DVB-H & T-DMB

In this chapter I will describe the two standards for broadcast television to handheld in Europe today, namely DVB-H and T-DMB, and what their differences and similarities are when it comes to broadcast television. Both systems originate from the idea that interactive television on handheld devices will be achieved by using existing technology such as GPRS or UMTS as the return channel.

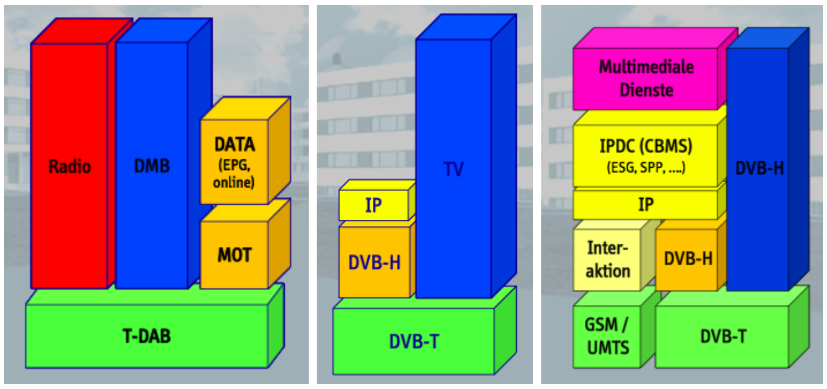


Figure 3.1: Block diagram of T-DMB, DVB-H and interactive DVB-H

By looking at figure 3.1 we can see the block diagrams of T-DMB, DVB-H, and a solution for interactive DVB-H with a return channel over UMTS. Subsequently, the DVB-H and the T-DMB features are described in detail.

3.1.1 DVB-H

DVB-H (Digital Video Broadcasting - Handheld) is a technical specification for bringing broadcast services to handheld receivers such as mobile phones, and it was adopted as an ETSI standard (EN 302 304) in November 2004. It uses real time-slizing, and has a bandwidth of up to 8 MHz, which makes room for up to 60 channels pr multiplex, according to the *System Comparison between T-DMB and DVB-H* [31]. DVB-H has three layers of error correction coding, convolutional, RS-FEC (ReedSolomon Forward Error Correction) and MPE-FEC (Multiprotocol Encapsulation Forward Error Correction), and has inner modulations of both QPSK [33], 16QAM and 64QAM [32]. The protocol stack for DVB-H is IP.

DVB-H adapts the DVB system for digital terrestrial television to handheld, battery powered terminals [3], and uses time-slicing technology to reduce total power consumption. Time-slicing enables the data packages to be transmitted by bursts in small time intervals, where each burst can contain up to 2 Mbits of data.

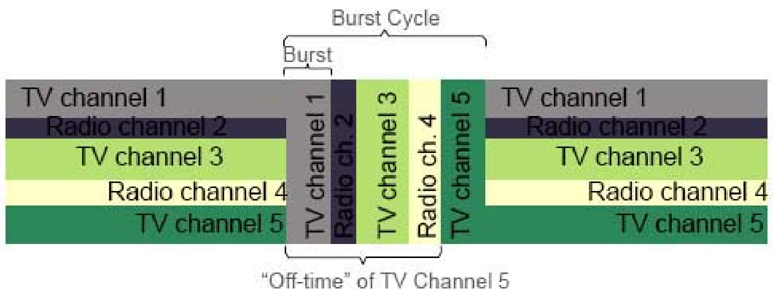


Figure 3.2: How Time-slicing works

IPDC

Another part of the DVB-H standard is IP datacasting (IPDC). Quoting the *IPDC Forum* [10],

> "IP datacasting is a broadcast technology which enables cost effective and efficient distribution of digital content to mass audiences."

This means that IP datacasting is a combination of digital broadcasting, IP based services and multimedia content, which enables a large scale distribution of multimedia content to the end-users. It is similar to the digital television broadcasting (DTV) system, with the difference that instead of channels being transmitted over a radio connection to a wide audience, data and files are sent. IPDC differs from the traditional Internet in the sense that content is not requested individually, but broadcasted to multiple receivers simultaneously. In theory, all content that can be carried through the Internet can also be broadcasted.

The variety of services that can be offered through IP datacasting is extensive. Most of the content that can be offered through the Internet can also be provided by using IP datacasting. Furthermore, by means of different communication channels, interactivity can be achieved in unidirectional networks. Therefore, content does not necessarily need to be broadcasted, it can also be requested, downloaded or streamed. According to *Texas Instruments* [9], some of the services that can be used are:

- Live television programs
- File downloads (music, images, ringtones)
- Electronic Program and service Guide
- Pay-per-view
- Packaged services (Sports events such as the World Cup)
- Live polls
- Chat

Services can be accessed when the end-users are on the move, in public transportation or transit, or just during the lunch break to catch up on the user's favorite sports teams, news, or special events such as the World Cup or the Olympics.

To provide a variety of services with the use of IP datacasting, several participants must cooperate in producing and managing the service for the mobile users. Figure 3.3 depicts the roles identified in the IPDC value chain:

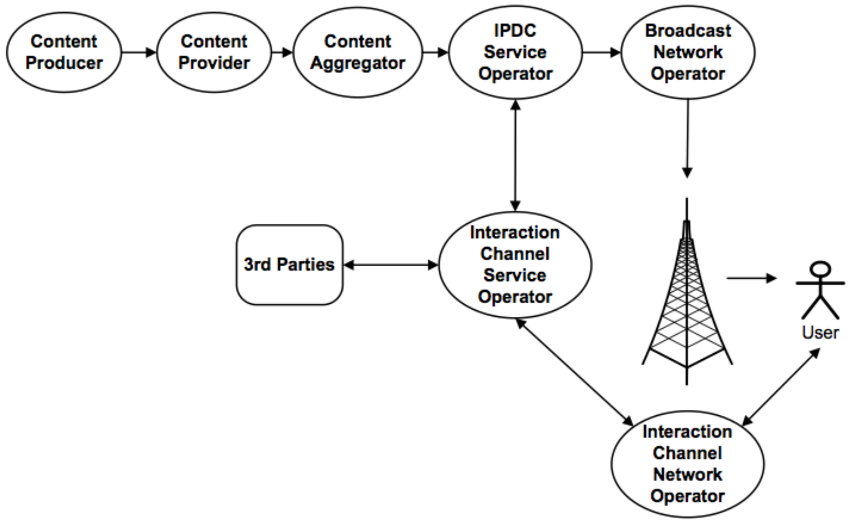

Content Producer
Content Provider
Content Aggregator
IPDC Service Operator
Broadcast Network Operator
3rd Parties
Interaction Channel Service Operator
User
Interaction Channel Network Operator

Figure 3.3: IPDC Value Chain

IP datagrams that are broadcasted over DVB-H are encapsulated inside the MPEG Transport Stream (TS) using Multiprotocol Encapsulation (MPE) to improve mobile performance. File delivery is performed by using File Delivery over Unidirectional Transport (FLUTE). FLUTE is a protocol for unidirectional delivery of files over the Internet [27]. Figure 3.4 describes the DVB-H system:

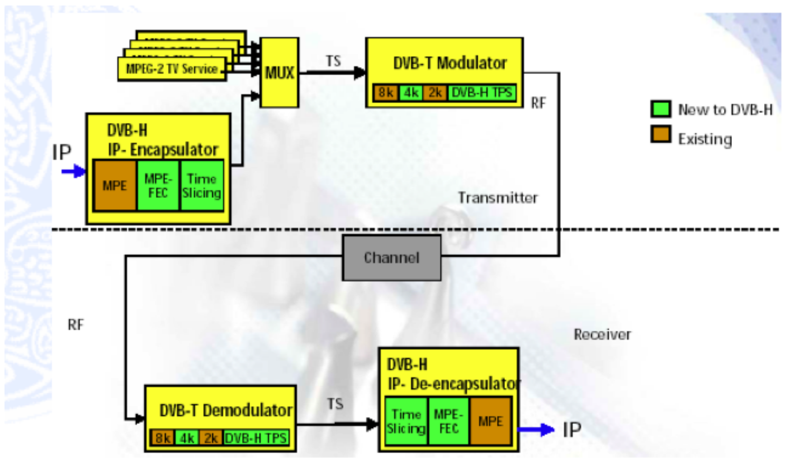
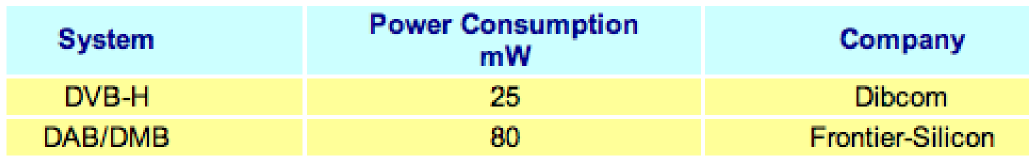


Figure 3.4: DVB-H Scheme

3.1.2 T-DMB

T-DMB (Digital Multimedia Broadcasting - Terrestrial) is a digital radio transmission system for sending multimedia content to handheld devices such as mobile phones, and was adopted as an ETSI standard (TS 102 427 and TS 102 428) in 2005. T-DMB uses micro time-slicing, and has a bandwidth of up to 1.712 MHz. Being narrowband T-DMB can hold up to 5 channels pr multiplex [31].

T-DMB is based on the Eureka 147 Digital Audio Broadcasting (DAB) standard [2], and has been modified for sending multimedia to handheld devices. T-DMB has two layers of error control coding, convolutional and RS-FEC. T-DMB has raw MPEG-4 protocol stack.



System	Power Consumption mW	Company
DVB-H	25	Dibcom
DAB/DMB	80	Frontier-Silicon

Table 3.1: Power consumption in mW for DVB-H and T-DMB systems

By looking at table 3.1, we can see that there is quite a difference between micro time-slicing (DMB) and real time-slicing (DVB-H).

Since T-DMB is based on the Eureka-147 system, T-DMB accommodates audio and data services as well as video services including MPEG-2 and MPEG-4 technologies [30].

For data services, T-DMB supports a variety of transport protocols such as Multimedia Object Transfer (MOT), IP Tunneling and Transparent Data Channel (TDC) [29]. For interactive service the MOT protocol is mainly used for the transmission of applications, and the TDC protocol is used for transmission of real-time streaming data. The BIFS (Binary Format For Scenes) in the MPEG4-standard is used for sending metadata, since DMB does not have an IP layer. As stated in *An efficient transmission framework of digital multimedia broadcasting (DMB) systems* [13], the DMB MPEG-4 data can be divided into two parts, the metadata and the media data. The metadata known as the IOD (Initial Object Descriptor), the OD (Object Descriptor) and the BIFS command data, and the media data as audio and video streams.

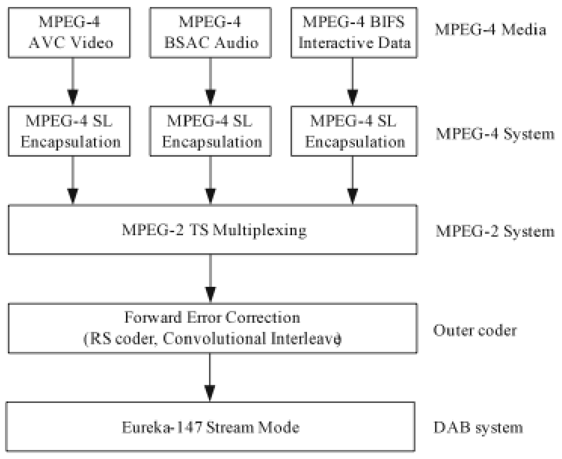


Figure 3.5: T-DMB Structure

Metadata Delivery Framework (MDF)

When considering the transmission of several media streams in a single stream mode sub channel without using the MPEG-2 system, several streams need to be multiplexed into one. According to *ETRI* [13], that can be done, and the result payload is called M4SMux,

> "which is designed for the stream mode transmission of MPEG-4 streams such as synchronously related audio-visual, clock reference, and BIFS-Animation stream."

This is one of the major differences between DMB and DVB-H, where DVB-H can simplify this prosess by using IP-packets. An example of how this is thought to be solved [13] can be seen in figure 3.6:

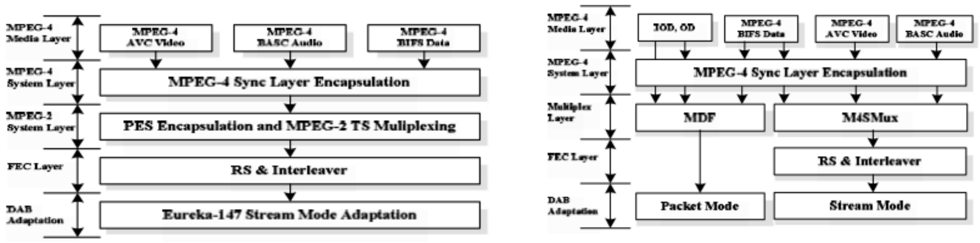


Figure 3.6: T-DMB architecture and proposed delivery architecture

3.2 Modulation

DVB-H and T-DMB both uses Orthogonal Frequency Division Multiplexing (OFDM) modulation. T-DMB uses an enhanced version of DAB's transmission system, with an additional layer of FEC coding to improve robustness and spectral effiency.

The following tables show the number of OFDM subcarriers that can be used for each of the systems, and the signal constellations that DVB-H and T-DMB can use to modulate the subcarrier. We can also see the number of bits per symbol that each signal constellation allows to be transmitted on each subcarrier.

System	DAB/DMB	DVB-H
Number of Subcarriers	1536	1705 (2K), 3409 (4K), 6816 (8K)

Table 3.2: OFDM subcarriers that can be used for DVB-H and T-DMB

Constellation	DAB/DMB	DVB-H	Bits / Symbol
QPSK	Yes	Yes	2
16-QAM	No	Yes	4
64-QAM	No	Yes	6

Table 3.3: Number of bits per symbol transmitted in DVB-H and T-DMB

As we can see from tables 3.2 and 3.3, using higher order signal constellations allows higher multiplex data rates, but it also increases the transmitter power.

T-DMB uses differential modulation, whereas DVB-H uses coherent modulation. An advantage with differential modulation is that the receiver only has to calculate the phase difference between the current and the previous subcarrier's phase angles, and it does not need to phase-synchronize. The disadvantage is that it incurs a 3 dB signal-to-noise (SNR) penalty relative to coherent modulation. Coherent modulation also allows 16-QAM and 64-QAM signal constellations to be used, which enables much higher bitrates and spectral effiency.

3.3 Interleaving

T-DMB uses both time- and frequency-interleaving, while DVB-H uses frequency-interleaving. DVB-H can on the other hand use time-interleaving in the 2K and 4K modes. Quoting the *system comparison T-DMB vs DVB-H* [31]:

> "DVB-H needs time-interleaving less than DMB due to DVB-H using significantly wider channels than DMB because at any instant in time the likelihood that a significant part of the channel bandwidth is in a deep fade is significantly less for DVB-H than DMB. DVB-H also has 2 outer layers of Reed-Solomon FEC coding, which are inherently very robust against burst errors."

3.4 Forward Error Coding Schemes

Apart from the modulation used, the FEC coding scheme is the most important part of a wireless digital communication system's transmission scheme. This means that stronger FEC coding schemes enables higher data rates, lower transmitter powers and more robust reception.

Figure 3.7 underneath shows the layers of FEC coding used in DVB-H and T-DMB:

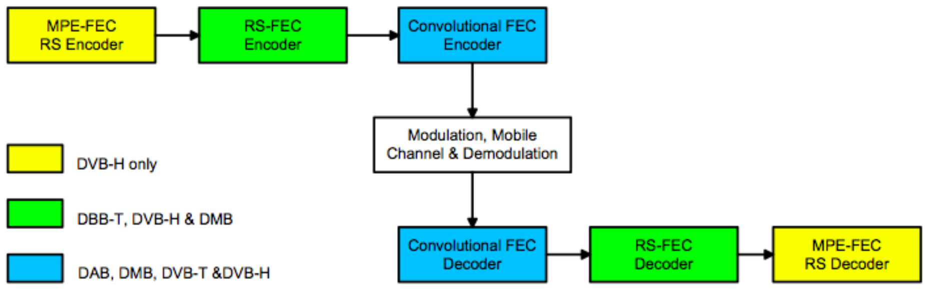


Figure 3.7: FEC coding layers in DVB-H and T-DMB

Reed-Solomon coding is used as the outer layer of FEC coding for T-DMB, and as middle layer of FEC coding for DVB-H. Using RS-code as the outer layer along with a convolution code for the inner layer of an FEC coding scheme is a very good combination for wireless systems. DVB-H goes even further by using another layer of FEC-coding, the MPE-FEC, which allows DVB-H to use 16-QAM and 64-QAM at reasonable C/N values at the receiver. MPE-FEC also allows DVB-H a much higher spectral effiency compared to T-DMB.

Part III

Own Contribution

Chapter 4

Interactive Services

The mobile phone has gone through major changes in the last decade. From being something purely made for short-time connections where you did not have access to a regular phone, it is today a device made to fit most purposes in everyday life. Some of these functions are music player, radio, game console, personal information management (PIM) and of course a small pocket-sized telephone.

The previous chapter described how the mobile phone can become a tool for interactive television, and the technologies behind that makes it possible to do so. To validate the concept, a demonstration is implemented. This chapter discusses the technologies behind such an implementation, and the technology used in the implemented demonstration.

4.1 Providing Interactivity

As the mobile phone has grown to suit more and more of our demands, there has been an increasing demand for more interactivity. Up until now the only interactivity has been *start* and *stop* on the media or music player, or if the user attended a game-show and sent a text message as a way of voting for his or her favorite player.

When receiving television over DVB-H or T-DMB, the user also gets metadata transferred to the handheld device. Having information about the tv-shows running at the moment, and also information about what kind of show it is, who stars in it, and how often it runs, an Electronic Program Guide (EPG) can be utilized. This will provide the end-user with much more interactivity since the end-user now can choose what kind of programs he or she would like to watch.

4.2 Use Case Scenario

In the use case scenario I will focus on how the the data and metadata received on the handheld device can provide the end-user with interactivity. If we consider someone watching a soccer game on their handheld device, he or she might be interested in more than just the game. Information about the teams, players, how they have done in the last few matches, voting for *Man of the Match* and perhaps even betting are all parts in the larger picture of a soccer game. Another important aspect is what time and date will these soccer games be aired? All of these examples can be realized by sending metadata along with the video stream, which again will be parsed within the handheld device and provides interactivity between content provider and end-user.

4.2.1 Seen By Client

From the end-user's perspective, watching a soccer game on his handheld device on his way to work can be a great way to spend time. The picture is formatted according to the resolution on the screen of the handheld, and he can do something useful (from his point of view) on his way to work. By the added interactivity due to metadata sent to him during the soccer game, he can now also get all the information he needs about his team and when their games will be aired, all of which he would have needed the internet for before. He stays updated on how many red or yellow cards each player has, if the player is injured or not, and he can also go online and bet on his team or voting for *Man of the Match* by just sending an text message back to the content provider. By voting for *Man of the Match*, he will be participating in a lottery of a signed jersey from his favorite team.

In this example, the user has selected soccer as one of his favorite sports, and Rosenborg as his favorite team. Therefore, the application filters the banners using the user's favorite team as parameter, mostly related to Rosenborg as a team and its players. By doing so, the banners are displayed to potential customers, since they are more likely to attract the user. A few examples here can be mentioned as:

- Informative banners, providing information about players and teams
- Purchase items such as wallpapers and ringtones.
- Web polls and live polls
- EPG showing when the desired soccer games will be aired on TV
- Third-party shopping channel
- Gambling
- Advertisement
- Voting and quizzes

These examples are just a few, but they all have in common that they provide the user with information and activities that makes the experience quite different from watching the soccer game at home. Now the user has the possibilities to actively take part in the game, hence participating in the interactive soccer game taking place on his mobile phone.

4.2.2 Seen By Provider

From the content provider's perspective there are a few more issues that need to be thought of. First of all the signal cannot be compromised, meaning that nobody will settle for less than a decent picture on their handheld screens. This means that a television broadcast must have a clear reception without any black spots and lag on the mobile screen.

On the positive side, interactivity with the end-user generates more money, because options like voting for *Man of the Match* is something many soccer fans favor. Since polls are quite easy to implement and manage, there can be live polls as well as regular web polls. A web poll is not dependant on the schedule of the TV show and can be shown at any time.

Advertisements and third-party shopping are elements that can create revenue for others than the content provider, as well as the content provider for providing it during shows. Gambling can also be mentioned here, since the betting companies make money off the users betting on their players, and the content and service providers get money from the text message sent by the end-user.

By using existing XML standard [45] the metadata will be formatted for show on a small-scale QVGA [23] screen, a format most hand-held device screens are today.

4.2.3 Example of an interactive soccer game

Figure 4.1 is an example of how interactivity can take place within a soccer game on a mobile phone.

Interactive features within a soccer game

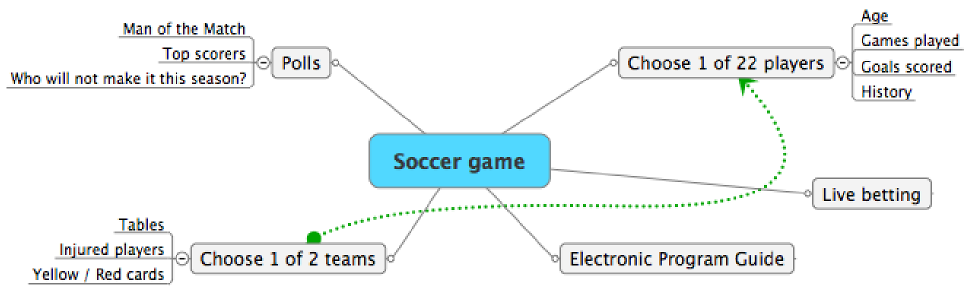


Figure 4.1: Interactivity within a soccer game

Underneath, in figure 4.2, the same interactivity functions are shown on a *Nokia* mobile phone.

Soccer game on a *Nokia* mobile phone

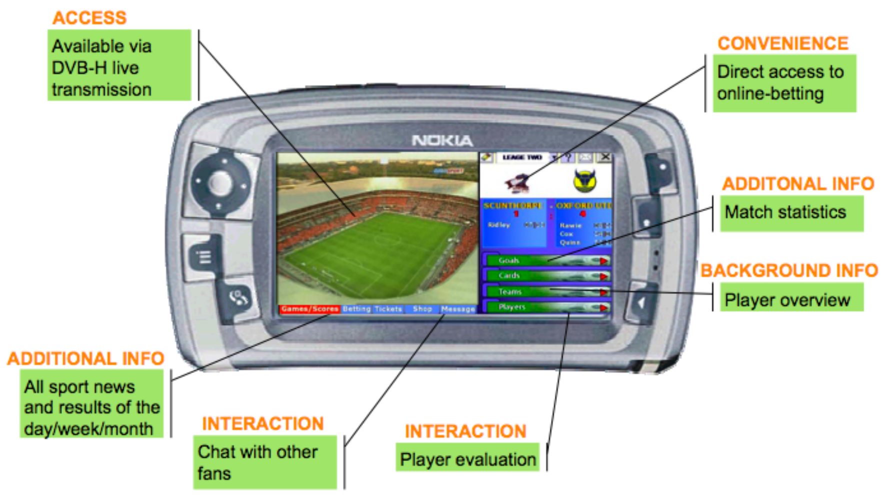


Figure 4.2: User interface showing an interactive soccer game on a *Nokia* mobile phone

4.3 User Control Issues

In this thesis I will show one way of how interactivity can be achieved by developing an interactive EPG to suit mobile phones that support the J2ME [22] enviroment. An EPG is the electronic version of a printed television program guide, especially made to suit a person *on the move*. It shows an on-screen guide to scheduled broadcast television programs, and allows the end-user to navigate, select and discover content by time, title, channel or genre.

To be able to create an EPG showing Norwegian television channels, the only feasible way at the moment is to involve a third party software solution called *XMLtv* [47] and combining the software with a script which has been custom-taylored to suit Norwegian TV channels, namely *tv_grab_no* [43]. XMLtv is a small program that grabs data from a source (usually a web-based program guide such as `http://fredag.dagbladet.no` or `http://dagenstv.com/no`, and converts it into an XML document. This XML document then makes up the feed for the mobile EPG application, and the EPG parses it to fit the screen of the mobile telephone.

Since interactive mobile television is quite different from regular television due to a smaller screen and the consumer being constantly *on the move*, four channels would be sufficient for the EPG in this thesis. This way the text will be readable, and the program will be compatible for most mobile phone displays. The program would be parsed for five days in advance, giving the consumer the possibility to browse forward and backwards in time for each channel.

The two main reasons for using an EPG is to stay updated on what kind of shows that run on each of the end-user's favorite channels, and when these shows are aired. To give the user an overview of what the shows that are running are about, metadata is sent along to give a brief synopsis about each show. Each channel are located on the Y-axis, and the X-axis is the timeline. This gives the best overview of all the channels at once, and the shows are marked as blocks that differ in length depending upon the length of each show. A red perpendicular bar across the timeline shows the present time. This way the user can browse backwards and forwards and get a quick overview of all the shows that are running. The user can navigate by using either the joystick or the buttons 2, 4, 6 and 8 on the numeric keypad. By hitting the key 5, the view will change into a short synopsis of that particular show. The user can also zoom in and out on the timeline to get a better overview of the whole day by using the "∗" and the "#" buttons on the keypad. This means that after configuring the sofware in means of which four channels the end-user wants, the only thing left to do to update the EPG is to log on to a server to get the latest XML document.

An EPG has two goals. One has been described above, namely to get an overview of which shows that are aired at one particular date, and when they are aired. The other goal is to be able to make a recording without spending much time adjusting and preparing the PVR in advance. This can be done by just implementing a box that can be checked if the user would like to record it. The recording would then be stored on a memory card or transferred to a larger storage unit. This falls outside the scope of this thesis, but is an important feature for EPGs on stationary SetTop Boxes used with digital TV in homes throughout the world today.

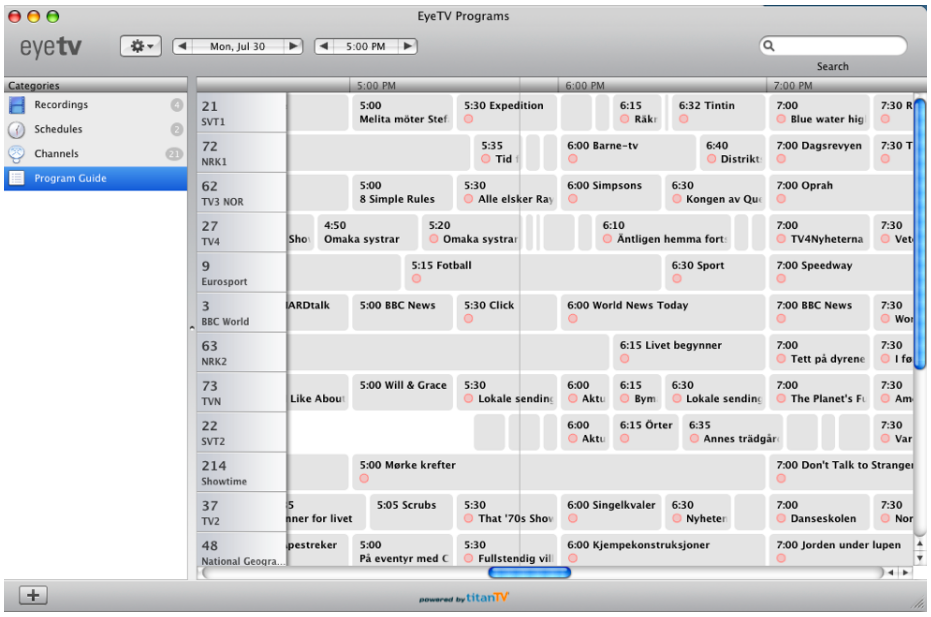


Figure 4.3: Picture of an EPG from a home SetTopBox

Figure 4.3 shows the EPG in Elgato Systems' *EyeTV* media center solution [6]. The red circle is the check-box to choose if you want to record the TV-show or not.

4.4 Application Logic Design

There are two different ways to implement an interactive EPG for handheld devices. Both of them are discussed in this section.

4.4.1 Hardware Application Approach

Interactive EPG could be implemented as an integrated addition to the handheld device from the manufactors such as *Nokia* or *SonyEricsson* themselves. A hardware application approach would help creating one world-wide standard for EPG on handheld devices. Since interactive handheld television is in its beginning phase at the moment, this solution has not been presented from any of the large mobile telephone manufactors yet. In the years to come, when TV receivers are more common parts of mobile phones, the solution might present itself to the consumers.

4.4.2 Software Application Approach

One solution to showcase an interactive EPG on a handheld device would be to develop a software application built upon the J2ME [22] enviroment. The application must rely on well-formed XML Schemas, which can be either in the form of DIDL elements as described in MPEG-21 [11], or in form of the XML standard [45]. When using *XMLtv* to generate the XML feed, the result is a valid XML Schema, but it contains white-spaces due to the HTML parsing, and therefore it would be time-saving when it comes to parsing the XML document in the client software if it has the form of the *Tv-Anytime* standard. This would then be added as a future extension to the mobile EPG created in this thesis, and happening on the server side. The mobile EPG *XMLguide* in this thesis is implemented on existing technology, and therefore uses the *XMLtv* format of its XML document.

There are a few issues connected to a software application approach however: The first issue is that Norway as a country has not been much involved with interactive television. Compared to countries such as USA and the UK, Norway is far behind in the development and use of interactive television. These countries have designated, corporate-driven websites constructed for only one purpose, namely to feed well-formed XML documents into the inhabitants personal video recorders (PVR) or TIVOs [41] for a small fee, creating a custom-taylored EPG of each end-user's choice. Two examples of sites mentioned above are *TitanTV* [40] in the US, and *Tvtv* [44] in the UK.

In Norway there is no such company or industry yet. Therefore, to be able to get the same result showing the Norwegian TV channels, the users are dependent upon scripts and software that parse HTML from websites such as `http://fredag.dagbladet.no` or `http://dagenstv.com/no` and formats the result according to the rules of a valid XML document. The most commonly used software for parsing HTML into XML is a third party software solution called *XMLtv* [47] and the script *tv_grab_no* [43], which has also been used in this thesis.

One of the major problems concerning parsing HTML into XML, is that whenever the websites used for parsing change in any way, the parser might not work properly. Since more and more people take advantage of features like interactive EPG in Norway and neighborhood countries, the scripts have improved quite a bit and only minor glitches might appear whenever this happens now. In the beginning, changes in the websites resulted in a crash for the *tv_grab_no* script.

4.5 Client-Server Architecture

The mobile EPG software works in two steps, first the third party software *XMLtv* grabs data from a web page (as mentioned above) and then it processes the data into an XML document named *mine4.xml* that can be parsed and viewed on the mobile phone display.

The optimal process would be to have a server doing parts of this process. The basic idea behind a server providing the EPG updates, is to reduce unnecessary use of bandwidth between the end-users and content provider. The server would be an additional instance between the content provider and the end-users, from where the end-users can obtain their data.

Having a server handling parts of the process, issues such as a change in content providers for EPG metadata will not affect the end-users in any way. The server is also able to convert the EPG metadata into new formats and standards to keep up with the software development and upgrades that would present itself in the future. An example here is to make the server retrieve the XML document on the *XMLtv* format, and then convert it into *Tv-Anytime* format before it is delivered to the end-users. This way the only thing the end-users would have to do, would be to log on to the server and retrieve the XML document to upgrade their EPGs. The XML document contains TV channel metadata for 5 days which then would be parsed into the EPG software running on the end-user's mobile phone.

Involving a server into this process also opens for possibilities such as logging and statistics of each end-user's preferences. This process has not yet been standardized, and could therefore be handled in numerous of different ways. Figure 4.4 shows the client-server architecture that is used in this thesis.

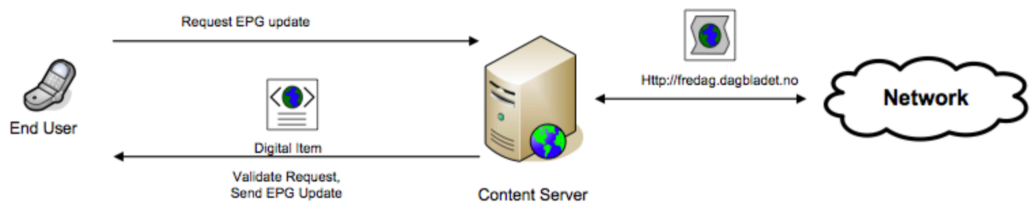


Figure 4.4: Request and delivery of a Digital Item EPG update

4.6 Implementation

There are two different parts of the implementation, and they are both discussed separately. The client side software is the interactive EPG specific part, and will be of primary focus in this thesis. Both the client side part and the server side part are needed to simplify the process of parsing XML data to ease the mobile CPU workload, and also to help the end-user stay updated on his or her four TV channels.

The client side is implemented in the Java programming language, and uses the Java Runtime Environment (JRE).

4.6.1 Client Side Implementation

Since none of the major mobile manufacturers have chosen their EPG software standards yet, there are a multitude of different ways of presenting XML metadata for the end-user. The solution in this thesis was to import some parts, such as the kXML parser [12], into an application programmed using the Java 2 Micro Edition (J2ME) [15]. The application was implemented using the open source platform Eclipse [4], and tested, simulated and debugged using the Java Wireless Toolkit for CLDC [17].

kXML

kXML is a small XML pull parser, specially designed for constrained environments and is therefore optimal to implement into our interactive EPG application. Why I would choose a pull parser rather than a push parser in a typical Connected Limited Device Configuration (CLDC) [14], is explained by the kXML website[12]:

> Pull based XML parsing combines some of the advantages of SAX (Simple API for XML) and DOM (Document Object Model):
>
> - In contrast to push parsers (SAX), pull parsers such as kXML make it possible to model the XML processing routines after the structure of the processed XML document. Events processing is similar to an InputStream. If a part of the stream requires special handling, the parser can simply be delegated to a specialized method by handing over the parser.
> - While the above is also possible with an explicit Document Object Model (DOM), DOM usually requires that the whole document structure is present in main memory.
> - In contrast to DOM based parsing, the XML events are accessible immediately when the XML tags are available, it is not necessary to wait for the whole tree to build up.

The latest version of kXML is based on the common *XmlPull API*. XmlPull is according to S. Haustein and A. Slominski [7] a simple, minimalist API:

> "XmlPull v1 API is a simple to use XML pull parsing API that was designed for simplicity and very good performance both in constrained environment such as defined by J2ME and on server side when used in J2EE application servers.

XML pull parsing allows incremental (sometimes called streaming) parsing of XML where application is in control - the parsing can be interrupted at any given moment and resumed when application is ready to consume more input."

XmlPull API		
int	`getEventType()`	Returns the type of the current event (START_TAG, END_TAG, TEXT, etc.)
java.lang.String	`getName()`	For START_TAG or END_TAG events, the (local) name of the current element is returned when namespaces are enabled
java.lang.String	`getAttributeValue(java.lang.String namespace, java.lang.String name)`	Returns the attributes value identified by namespace URI and namespace localName
java.lang.String	`getText()`	Returns the text content of the current event as String

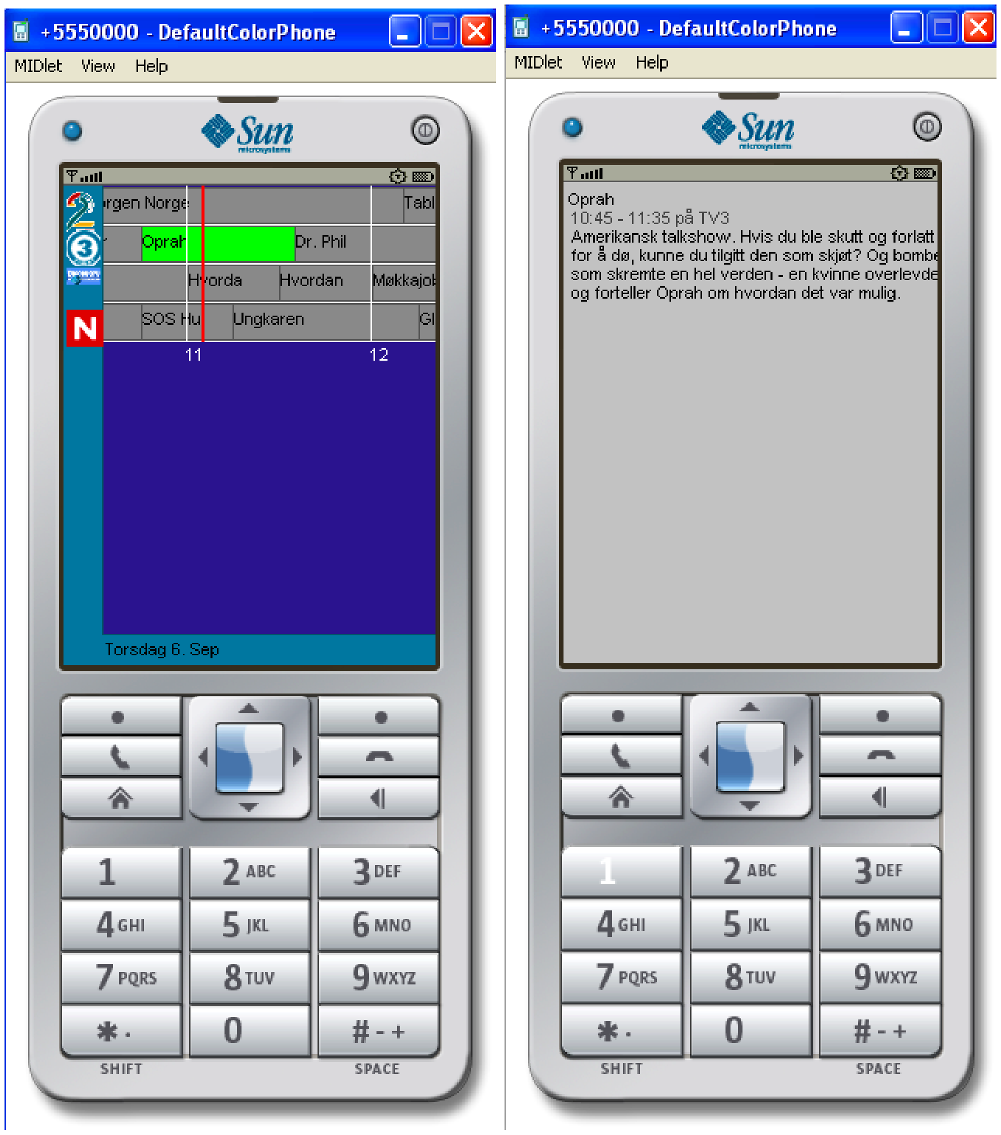
Table 4.1: kXML API methods

The aim of kXML is to provide a similar but orthogonal pull parsing basis to widely successful push parsing SAX API. The API that the XmlPull implements is given in table 4.6.1. There are quite a few reasons why the XmlPull v1 API has become so popular, and according to S. Haustein and A. Slominski [7] a few keywords can be mentioned:

- **Simple interface** - parser consists of one interface, one exception and one factory to create parser
- **Implementation independent** - factory class is modeled after JAXP [18] and allows easily to switch to different XmlPull v1 API implementation without even modifying source code
- **Ease of use** - there is only one key method next() that is used to retrieve next event and there are only five events:
  - **START DOCUMENT** - document start - parser has not yet read any input
  - **START_TAG** - parser is on start tag
  - **TEXT** - parser is on element content
  - **END_TAG** - parser is on end tag
  - **END_DOCUMENT** - document finished and no more parsing is allowed
- **Versatility** - it is generic interface for XML parser and allows for multiple implementations and extensibility through features and properties
- **Performance** - the interface is designed to allow implementing very fast XML parsers
- **Minimal requirements** - designed to be compatible with J2ME (Java 2 Micro Edition) to work and on small devices and to allow create XmlPull compliant parsers of very small memory footprint.

XMLguide

The program *XMLguide* is an application programmed using the Java 2 Micro Edition (J2ME) [15] that parses XML, and outputs the result to a Graphical User Interface (GUI) on the end-user's mobile phone. The result is a custom-taylored TV-guide showing four channels within a time-frame of five days. For a more thorough explanation of the EPG, please see section 4.3.



Oprah
10:45 - 11:35 på TV3
Amerikansk talkshow. Hvis du ble skutt og forlatt
for å dø, kunne du tilgitt den som skjøt? Og bombe
som skremte en hel verden - en kvinne overlevde
og forteller Oprah om hvordan det var mulig.

Figure 4.5: Picture of EPG and metadata for the show *Oprah*

Figure 4.5 shows how the EPG would look on a mobile telephone for the channels TV2, TV3, Discovery and TvNorge. The second image shows the metadata for the highlighted show *Oprah*.

XmlGuide consists of 6 classes. These classes are *MainClass, GuideCanvas, ProgramClass, ChannelClass, XmlGuideParser and st*, and I will give a brief summary of what each class does. For further information, please see appendix C.

- **MainClass** initializes the other classes and instantiates GuideCanvas, which is the main object in the program.
- **GuideCanvas** draws the canvas and creates the GUI that the end-user sees on the mobile phone. GuideCanvas also handles everything that happens within the GUI, from navigating through each program within each channel, getting the synopsis and redrawing the picture whenever needed. To simplify the debugging process, dummy channels have been used to first get the process right. These are still inside the program code, but sorted out by a true-false statement where true would initialize the XML feed instead of loading the dummy channels. The canvas adjusts itself to the mobile screen and creates a buffer to avoid flicker on the screen. What also is worth mentioning is that GuideCanvas also converts the dates from *string* to *unitime* and stores it as a *long int*. This means that the program will overflow after year 2030, but works fine until then. (By that time there *will* be an upgrade of any sort available).
- **ProgramClass** holds the information about one program in a string. This means the time it starts, the time it ends, the name of the program and its description.
- **ChannelClass** holds the information about one channel. One channel can contain a variety of different programs. ChannelClass is also told to discard programs that are too short, which means less than 5 seconds in length.
- **XmlGuideParser** parses XML into data objects. This class imports kXML and makes use of existing technology for parsing XML in a CLDC [14] enviroment. XmlGuideParser parses the XML document *mine4.xml*, which consists of the channel programs and their metadata.
- **st** is a class with all the static global variables inside. This was done to make it easier to debug the program.

In addition to initializing the classes, an important function within XmlGuide is *PaintBoard*, which draws the buffer before it is released to the mobile screen. Repaint will be called upon whenever necessary to redraw the canvas when navigating around or zooming in or out. It can also be mentioned that every timestamp in the XML document had to be converted to pixels on the mobile phone display and vice versa to get the right width of each program.

4.6.2 Server Side Implementation

To automate the process of obtaining the XML metadata feed, the entire *XMLtv* process can be done by a server and then made available for download. This way the end-user only has to log on to the server address, which in this case would be `http://folk.ntnu.no/andrease/mine4.xml`, from his or her mobile phone to update the EPG. There are a multitude of different protocols and techniques that can be used to implement such a server. The easiest solution would be making a script for requests and transmission by TCP/IP HTTP protocol. PHP programming language could be utilized in combination with Apache web server to generate an XML document for the end-users. This way the end-users could just log on to the server with the web browser on their mobile phones and retrieve the XML document to upgrade their EPGs. Specifications such as channel selection and number of days could be handled by sending parameters from the client during the request. The server would then check if the requested data already resides within cache, and if not repeat the entire process before sending the XML document back as an answer to the request.

In this thesis, the process was done by using the feature *Scheduled Tasks* from a windows server, and uploading the XML file to the url `http://folk.ntnu.no/andrease`.

Part IV

Evaluation and Conclusion

Chapter 5

Evaluation

5.1 Discussion

I have in this thesis focused on how interactivity on handheld devices can be achieved by combining broadcasted television with metadata formatted in accordance with the XML [45] standard. Both of the standards for handheld television in Europe today, DVB-H [21] and DMB [24], have been included in this analysis, as well as third party software solutions that enables interactivity between the end-user and content provider. The essence of interactive handheld television is, according to *NTT Cyber Solutions Laboratories* [36]:

> "A typical mobile user is expected to view content not in long stretches but selectively, watching only items of interest over short time spans. To enable users to view only desired content of a short duration in a limited time, we must provide some means of matching content to individual preference attributes."

Interactive handheld television is not a competitor of the normal way of watching television. Instead it should provide an added-value to the user. Mobile television should provide the user with a different experience than watching television at home. Mobile telephones are personal and therefore have to adapt to the user's behavior and needs. The same is also true with mobile television. Furthermore, it should give the user the opportunity to interact with others that share the same interests.

To be able to match content to individual preference attributes, the content provider is dependent upon having some sort of return channel to create interactivity and feedback. By using existing mobile technology structure, GSM, EDGE and UMTS can be used as the return channel and make way for new business models to be developed. Offering customized content and services through different channels can provide the means for further revenues. The mobile terminal can then act as a portal, accessing both TV and different kinds of services within the same application.

As the demonstration implemented in this thesis shows, obtaining interactivity through an Electronic Program Guide (EPG) on a mobile phone is feasible. It also shows that the television watching experience changes when adding interactivity to it, hence making mobile television quite attractive to the consumers. Whether the content providers charge too much for this experience, will become an important question to ask when TV-receivers in the future are added as a standard addition to the mobile phone.

As we can see from figure 5.1, the combined solution of both cellular- and broadcast-network [1] can be used to custom-taylor interactive television content for the end-user. The picture shows how *Nokia* plans to implement mobile television over DVB-H as a solution to their mobile phones.

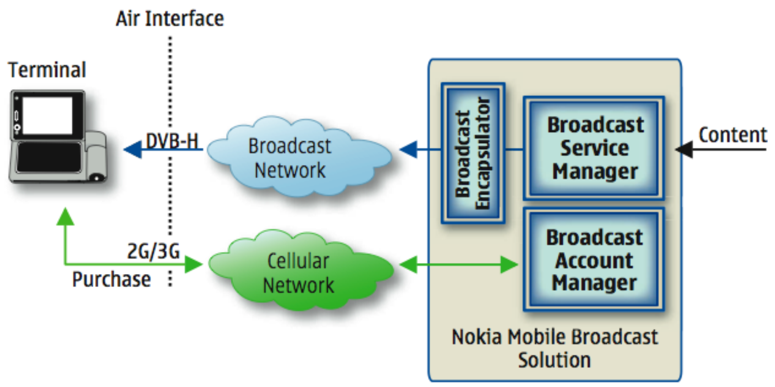


Figure 5.1: How interactive television can be achieved on a *Nokia* mobile phone

Combining broadcast streaming with metadata delivered in accordance with the XML standard [45], the interactive handheld mobile is not restricted to any of the two specific standards for mobile broadcast television. The Tv-Anytime standard [42] outlines a solution to how the details of the EPG's metadata, rights management and access control can be managed for DVB-H broadcasts, and being sent alongside the broadcast stream, the metadata would come at no extra cost. With metadata sent alongside the DVB-H broadcast, no server would be needed for the EPG software application, and everything would be done on the client side. Unfortunately this has not happened in Norway yet.

In the last year development of mobile interactivity has escalated, and many companies have found their niche in this new and expanding field of innovation. The Norwegian company *Adactus* (`www.adactus.no`), which I have had a close relationship with during this thesis, is one of these. Adactus has implemented an interactive application for mobile phones for the Norwegian internationally known company *Telenor* (`www.telenor.no`).

[1]The picture is taken from a DVB-H setting, but here it is a symbol of broadcasted television, and can be used for both T-DMB and DVB-H

Nokia has also seen the potential revenue in this marked, and has been developing their own versions of a similar application. Underneath interactive mobile television as a whole can be seen in figure 5.2. The picture outlines how *Nokia* plans to implement interactive broadcast over DVB-H, by adding a server to manage voting by text messages as a way for the end-user to interact with the content provider.

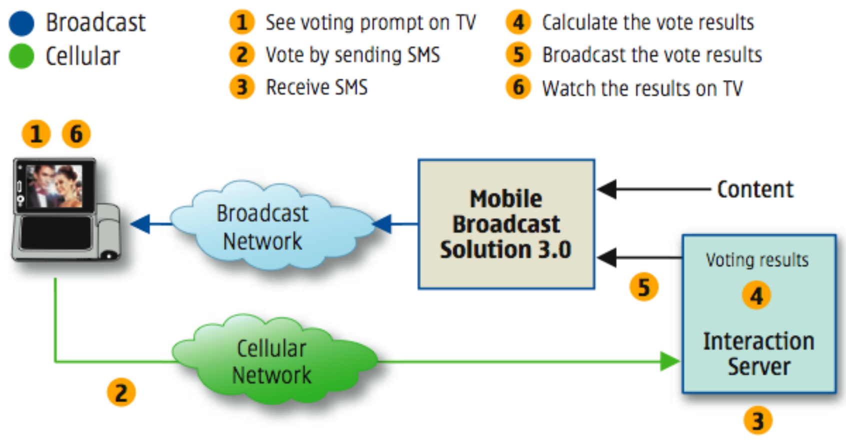

Broadcast
Cellular
1 See voting prompt on TV
2 Vote by sending SMS
3 Receive SMS
4 Calculate the vote results
5 Broadcast the vote results
6 Watch the results on TV
1 6
Broadcast Network
Mobile Broadcast Solution 3.0
Content
Voting results
4
Interaction Server
5
Cellular Network
2
3

Figure 5.2: Interactive mobile television seen from *Nokia's* perspective

Chapter 6

Conclusion

6.1 Conclusion

As I set out on this project, my intentions were to see what I could find out about television on handheld devices today, and what needs to be done to enable interactivity between the end-user and the content provider.

We have seen that DVB-H and DMB provide the bearer technologies for handheld television. However, in order to deliver a service the end-user finds acceptable, service layer components are required on top of this. Audio and video codecs are optimized through the MPEG-standards, and MPEG-4 makes the content scalable for QVGA screens. XML, MPEG7 and MPEG-21 DIDL allow metadata to be formatted and shown as Electronic Program or service Guide (EPG) and interactivity to be achieved.

In addtition to this, a service and content protection system is required in order to support pay services and prevent content piracy and fraud. Middleware that allows broadcasted, interactive television to work on multiple terminals is also needed to make interactive mobile television come true. These spesifications are currently being developed.

When it comes to which technology is *best* suitable for interactive television on handheld receivers, the answer is quite complex. In short, the overall relationship between T-DMB and DVB-H is that T-DMB is a small-scale system which is cost efficient when the total demand for broadcasting capacity is small. T-DMB can handle up to 5 channels and has a bandwidth of approxmently 1.5 MHz, whereas DVB-H on the other hand is a large-scale system which will be more cost-efficient when the demand for capacity is large. DVB-H can have up to 60 channels pr multiplex and a bandwidth up to 8 MHz. Both technologies work well according to the test pilots in their respective countries, and each of them have pros and cons.

DVB-H has real time-slicing as a crucial component, which offers up to 90 % battery saving, and it also has an IP layer which enables IPDC. DVB-H also has larger bandwidth, and according to the conclusions of the *System comparison between DVB-H & DMB* [31], it is more future-oriented.

The demonstration provided in this thesis shows that creating interactivity on a mobile phone by using metadata on the XML format to feed an Electronic Program Guide (EPG) is feasible. The application can be used on any mobile telephones that support the J2ME framework.

6.1.1 Recent Development

New and promising projects are under development, and especially one called Digital eXtended Broadcasting (DXB) [37] is interesting with regards to this thesis. The DXB project group is working on adding an IP layer and time-slicing to the T-DMB structure. By doing so they are creating a small-scale DVB-H-like concept, which adds the best part from DVB-H and Multimedia Broadcast Multicast Service (MBMS) [20] into the concept:

> "DVB-H and T-DMB are technically similar at the lower physical layer since both use the transmission technique COFDM (Coded Orthogonal Frequency Division Multiplexing). More commonality could possibly be found by developing a common protocol stack, leading to a simplified implementation in common chipsets. Such a "combined DXB approach" with DVB-T-based and DAB-based physical layers could ultimately lead to significant economes of scale and lower consumer costs [1]."

Recent development in 2007 has shown that in Finland, the software development company *Sofia Digital* [38], has in collaboration with *Nokia* launched a complete Electronic Program Guide (EPG) solution that receives metadata on the Tv-Anytime standard over DVB-H. They launched their EPG solution *Sofia Backstage*® on September 6th, and what the Vice President, Interactive TV from Sofia Digital told the press tells us that interactive mobile television is here to stay:

> "Mobile TV offers a good environment for various interactive services because the cellular networks offer a natural return-channel for using the services. Our expectations in rolling out the services are to help TV channels to offer a refined and full viewer experience also to their mobile TV audience"

[1]Quoting the EBU Technical Review, January 2006 [46]

Bibliography

[1] Ian S. Burnett, Fernando Pereira, Rik Van de Walle, and Rob Koenen. *The MPEG-21 Book*. John Wiley & Sons, Ltd, 2006.

[2] DAB. http://en.wikipedia.org/wiki/Digital_Audio_Broadcasting, 2006.

[3] DVB. http://en.wikipedia.org/wiki/DVB, 2006.

[4] Eclipse. http://www.eclipse.org, 2007.

[5] Elgato-Systems. Eyetv. http://www.elgato.com/index.php?file=products_whatiseyetv, 2007.

[6] Elgato-Systems. Media center solutions for mac. *EyeTV*, 2007.

[7] Stefan Haustein and Aleksander Slominski. Xml pull parsing. http://xmlpull.org/, 2006.

[8] iFanzy. http://en.wikipedia.org/wiki/IFanzy, 2007.

[9] Texas Instruments. Dvb-h mobile digital tv for the u.s. *Texas Instruments Incorporated*, 2005.

[10] IPDC-Forum. Ipdc. http://www.ipdc-forum.org/, 2006.

[11] Joseph Thomas Kerr, Ian Burnett, and Philip Ciufo. Bitstream binding language: Mapping xml multimedia containers into streams. *Smart Internet Technology*, 2005.

[12] kXML. http://kxml.sourceforge.net/about.shtml, 2005.

[13] Bong-Ho Lee, So Ra Park, Young Kwon Hahm, and Soo In Lee. An efficient transmission framework of digital multimedia broadcasting (dmb) systems. *Electronics and Telecommunications Research Institute*, 2004.

[14] Sun Microsystems. Connected limited device configuration (cldc). http://java.sun.com/products/cldc/, 2007.

[15] Sun Microsystems. Java 2 micro edition. http://java.sun.com/javame/index.jsp, 2007.

[16] Sun Microsystems. Java virtual machine. http://en.wikipedia.org/wiki/Java_Virtual_Machine, 2007.

[17] Sun Microsystems. Java wireless toolkit for cldc (jwt). http://java.sun.com/products/sjwtoolkit/, 2007.

[18] Sun Microsystems. Jaxp. http://java.sun.com/webservices/jaxp/index.jsp, 2007.

[19] MPEG. http://en.wikipedia.org/wiki/MPEG, 2006.

[20] Multiple. Multimedia broadcast multicast service (mbms). *Wikipedia*, 2004.

[21] Multiple. Dvb-h. http://www.dvb-h.org/, 2006.

[22] Multiple. Java platform, micro edition. http://en.wikipedia.org/wiki/Java_ME, 2006.

[23] Multiple. Quarter video graphics array (qvga). http://en.wikipedia.org/wiki/QVGA, 2006.

[24] Multiple. T-dmb. http://eng.t-dmb.org/, 2006.

[25] Multiple. World wide web consortium. http://www.w3.org/, 2006.

[26] Multiple. Xml schema. http://www.w3.org/XML/Schema, 2006.

[27] T. Paila, Nokia, M. Luby, Digial Fountain, R. Lehtone, TeliaSonera, and V. Roca. Flute: File delivery over unidirectional transport. *IETF RFC 3926*, 2004.

[28] Andrew Perkis, Yousri Abdeljaoued, Charilaos Christopoulos, Touradj Ebrahimi, and Joe F. Chicharo. Universal multimedia access from wired and wireless systems. *Circuits, systems and signal processing*, 20(3):387–402, 2001.

[29] Project. Digital audio broadcasting (dab); multimedia object transfer (mot) protocol. Technical report, ETSI EN 301 234 v1.1.1, 1999.

[30] Project. Radio broadcasting systems, digital audio broadcasting (dab) to mobil, portable and fixed receivers. Technical report, ETSI EN 300 401 v1.3.3, 2001.

[31] Digital Video Broadcasting Project. System comparison t-dmb vs. dvb-h. Technical report, Digital Video Broadcasting Project, 2006.

[32] QAM. http://en.wikipedia.org/wiki/Quadrature_amplitude_modulation, 2006.

[33] QPSK. http://en.wikipedia.org/wiki/Phase-shift_keying, 2006.

[34] Uwe Rauschenbach. Interactive tv meets mobile computing. *Siemens AG, Corporate Technology*, 2005.

[35] Isaac Richards. Mythtv. http://www.mythtv.org/, 2002.

[36] Takeshi Sasaki, Masahito Kawamori, Jin Hatano, and Katsuhiko Kawazoe. Using metadata to control content viewing on mobile devices. *NTT Cyber Solutions Laboratories*, 3(8):47–51, Aug 2005.

[37] Ralf Schaefer. Dxb ein harmonisierter ansatz fuer mobile tv. *Fraunhofer HHI*, 2006.

[38] Sofia-Digital. Tv-anytime based epg over dvb-h. http://www.sofiadigital.com/content/view/109/50/, 2007.

[39] Peter Stollenmayer. Mobile tv - the next european mega trend? *EURESCOM mess@ge*, 2:7–8, 2006.

[40] TitanTV. Tv listings based on your zip code. http://www.titantv.com, 2007.

[41] TiVo. http://en.wikipedia.org/wiki/TiVo, 2007.

[42] Tv-Anytime-Forum. Tv-anytime. http://www.tv-anytime.org, 2006.

[43] tv_grab_no. Grab tv listings for norway. http://www.penguin-soft.com/penguin/man/1/tv_grab_no.html, 2005.

[44] Tvtv. Everything else is just tv. http://www.tvtv.co.uk, 2007.

[45] W3C. Xml. http://www.w3.org/XML/, 2006.

[46] Chris Weck and Edgar Wilson. Broadcasting to handhelds - an overview of systems and services. Technical report, EBU, 2006.

[47] Xmltv. http://xmltv.org/wiki/, 2007.

Part V

Appendices

Appendix A

On sources

In addition to the cited sources in this thesis, I have used pictures and illustrations from *Wikipedia* to illustrate the chapter on technical data about DVB-H & T-DMB. The pictures used are released either under a Creative Commons license or into the public domain.

In addition, I have read a large number of articles on DVB-H, T-DMB, XML, XMLtv, J2ME, Tv-Anytime and the different MPEG standards, articles that have not necessarily been used as direct sources in this report but that has nevertheless helped me gain a body of knowledge that has been a tremendous asset in performing the study.

Appendix B

List of analysed articles

Below is a list of all articles that was analysed, along with the authors and the year they were published:

L. Stomback *A classification for comparing standardized XML data* 2006

M. Kim *Agent-Based Intelligent Multimedia Broadcasting within MPEG-21Multimedia Framework* 2004

D. Skiold *An Economic Analysis of DAB & DVB-H* 2006

S. Ioannou et al. *Archiving Multimedia Content Descriptions: An Early Adaption of MPEG-7* 2001

F. Sommers *Broadcast Once, Watch Anywhere* 2005

DigiTAG *Broadcasting to Handheld Devices* 2005

DVB Fact Sheet *Broadcasting to Handhelds* 2007

Digitalradiotech *Comparison of DAB, DMB & DVB-H Systems* 2006

P. Pogrzeba T-Systems *Challenges of IP Datacast Business* 2005

B. Bae et al. *Design and Implementation of the Ensamble Remultiplexer for DMB Service* 2004

G. Lee et al. *Design of Middleware for Interactive Data Services in Terrestrial DMB* 2006

ETSI *Digital Audio Broadcasting (DAB); Data Broadcasting - MPEG-2 TS streaming* 2005

ETSI *Digital Audio Broadcasting (DAB); DMB video service; User Application Specification* 2005

ETSI *Digital Audio Broadcasting (DAB); DMB Video Service; User Application Specification* 2005

ETSI *Digital Audio Broadcasting; Transportation and Binary Encoding Specification for EPG* 2005

ETSI *Digital Audio Broadcasting; XML Specification for EPG* 2005

ETSI *Digital Video Broadcasting (DVB); DVB-H Implementation Guidelines* 2005

NOKIA *Digital Video Broadcasting, Handheld (DVB-H) - Explaining DVB-H*2007

ETSI *Digital Video Broadcasting (DVB); Transmission to Handheld Terminals (DVB-H)* 2005

ETSI *Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H)* 2004

NOKIA *Digital Video Broadcasting, Handheld (DVB-H) - Press Backgrounder* 2005

NOKIA *DVB-H - Delivering the TV experience to mobile devices* 2006

G. Faria et al. *DVB-H: Digital Broadcast Services to Handheld Devices* 2006

NOKIA *DVB-H; Made for Mobile Devices* 2005

M. Kornfeld et al. *DVB-H - the emerging standard for mobile data communication* 2005

Texas Instruments - white paper *DVB-H mobile digital TV* 2006

J. Henriksson *DVB-H General Outline* 2003

Dr. C. Sattler *DVB-H Pilot Trial in Berlin, EBU Forecast* 2004

Texas Instruments *DVB-H mobile digital TV for the U.S* 2005

H. Twietmeyer *DMB & DXB: Multimedia via DAB Limits and Potentials* 2005

Department of Trade and Industry *Digital Television Project: Provision of Technical Assistance* 2003

Norkring *Den digitale kringkastingsrapporten* 2005

A. Hickman *Enhancing TV-Anytime with Metadata from Bi-Directional Channel* 2003

R. Troncy et al *Enabling Multimedia Metadata Interoperability by Defining Formal Semantics of MPEG-7 Profiles* 2006

Phillip Research Laboratories et al. *Extensibility of metadata Response to CfC TV 08* 2001

Cap Gemini, Telecom, Media & Entertainment *Exploiting the Growing Content Value Chain* 2006

Phillip Research Laboratories et al. *Extensibility of metadata in TV Anytime* 2001

IST Project Telefonica *ePerSpace Services* 2005

P. Bristov et al. *GEM-IPTV white paper* 2004

J.H Kim et al. *Implementation of Chipset for a Digital Multimedia Broadcasting Receiver* 2005

ETRI *Interactive Data Sercies on T-DMB Using MPEG4-BIFS* 2006

K. Kim et al. *Interactive Data Services on T-DMB* 2005

Y. Parker *Integrated digital medium: Convergence of digital tv, web and mobile communications* 2002

DVB Fact Sheet *Internet Protocol Datacast - Making Mobile TV Happen* 2007

Sonera MediaLab *IP Datacasting Content Services - White Paper* 2003

F. Schreiner et al. *ISO/IEC JTC1/SC29/WG11MAF Overview* 2007

A. Karanastasi *Language model for managing TV-Anytime information in mobile enviroments* 2004

C. Morecraft *Metadata - The Role of the TV-Anytime Specification* 2004

G. Martinez *Mobile Broadcast Standards - Focus on DVB-IDC and OMA-BCAST over DVB-H* 2006

S. Meltzer et al. *MPEG-4 HE-AAC v2 -audio koding for today's digital media world* 2006

ISO/IEC JTC *MPEG-21 Digital Item Declaration WD (v2.0)* 2001

S. van Assche et al. *Multichannel Publication using MPEG-21DIDL and extensions* 2003

Dr. U. Reimers *Multimedia on the Move: Technologies for Broadcasting to Handheld Devices* 2006

A. McParland et al. *MyTV: A Practical Implementation of TV-Anytime on DVB and the Internet* 2002

Phillip Research Laboratories et al. *MyTV Content Referencing: Frequently Asked Questions* 2001

Phillip Research Laboratories et al. *MyTV Proposal on "A system's view on metadata"* 2001

K. Millar et al. *NDS Contribution from MyTV, A Schema for TV-Anytime: Segmentation Metadata* 2000

L. Viken et al. *Norges Televisjon AS; Digitalt bakkenett i Norge* 2005

Silicon & S. Systems Ltd *onHandTV Datasheet* 2007

RadioScape Ltd *Professional DAB/DMB Broadcast System* 2006

NSK NewsML Team *ProgramGuideML Specification Verson 1.0 Beta* 2004

Phillip Research Laboratories et al. *Proposal for MPEG-2 Carriage of Content Referencing Information* 2001

P. Lieu et al. *Queries of Digital Content Descriptors in MPEG-7 and MPEG-21 XML documents* 2003

M. Petersen *Semantic extension of TV-Anytime metadata in DVB-H mobile broadcast* 2006

M. Petersen *Semantic media description in converged DVB-H and 3G* 2006

DVB Org *Specification for use of Video and Audio Coding in DVB Services delivered over IP Protocols* 2005

Tv-A. Forum *Spesification Series: S-3 on: Metadata (Normative)* 2001

Tv-A. Forum *Specification Series: S-4 on: Content Referencing (Normative)* 2001

DigiTAG *Television on a handheld receiver - broadcasting with DVB-H* 2005

T. Multimedia inc *T-DMB Middleware Sulution for Mobile TV* 2006

Electronic & Telecommunication Research Institute (ETRI) *T-DMB White Paper* 2006

Klaus Pilz *TV Goes Mobile* 2005

J.P Evain et al *TV-Anytime Phase 1 - A Decisive Milestone in Open Standardisation for PVRs* 2004

S. Pfeiffer et al *TV-Anytime as an application scenario for MPEG-7* 2000

R. M. Tol et al *TV-Anytime: Store it on my TV* 2000

R. S. Atarashi (CRL) *Use of Metadata in XML Configuration* 2004

M. Pogacnik *User modeling based on TVAnytime metadata standard* 2003

Dr. K. Illgner *What is needed for Mobile Broadcast Services?* 2005

F. S. Wendelstein *Why DMB is an alternative to DVB-H in Germany* 2005

M. Cokus et al. *XML Binary Characterization Use Cases* 2004

Appendix C

XML and J2ME code

Listing C.1: Soccer game example - *SoccerGame.xml*

```xml
<?xml version="1.0" encoding="UTF-8"?>
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:mp7="http://www.mpeg7.org/MP7Schema"
  xmlns:tva="http://www.tv-anytime.org/TVASchema"
  targetNamespace="http://www.tv-anytime.org/TVASchema"
  elementFormDefault="unqualified" attributeFormDefault="unqualified">
<Declarations>
  <Descriptor id="copyright">
    <Statement mimeType="text/plain">
  </Descriptor>
</Declarations>
<Container id="Sports_channel">
  <Item id="Soccer_game">
    <Descriptor>
      <xi:include xpointer="copyright/1"/>
    </Descriptor>
    <Descriptor>
      <Statement mimeType="text/plain">
        Soccer game between Norway and Sweden
      </Statement>
    </Descriptor>
    <Descriptor>
      <Component>
        <Resource mimeType="image/jpeg" ref="http://server/no_swe_flag.png"/>
      </Component>
    </Descriptor>
    <Choice choice_id="laginfo" minSelections="1" maxSelections="1">
      <Descriptor>
        <Statement mimeType="text/plain">
          Do you want information about the playing teams?
        </Statement>
      </Descriptor>
      <Selection select_id="no_information">
        <Descriptor>
          <Statement mimeType="text/plain">
            I do not want information about the teams!
          </Statement>
        </Descriptor>
      </Selection>
      <Selection select_id="information">
        <Descriptor>
          <Statement mimeType="text/plain">
            I want information about the teams
          </Statement>
        </Descriptor>
      </Selection>
    </Choice>
    <Choice choice_id="format" minSelections="1" maxSelections="1" default="tables">
      <Condition require="information" />
      <Descriptor>
        <Statement mimeType="text/plain">
          What do you want to know about the teams?
        </Statement>
      </Descriptor>
      <Selection select_id="tables">
        <Descriptor>
```

```
          <Statement mimeType="text/plain">
            Statistics
          </Statement>
        </Descriptor>
      </Selection>
      <Selection select_id="player_information">
        <Descriptor>
          <Statement mimeType="text/plain">
            I want information about the players
          </Statement>
        </Descriptor>
      </Selection>
      <Selection select_id="injured_players">
        <Descriptor>
          <Statement mimeType="text/plain">
            I want to see who is is injured
          </Statement>
        </Descriptor>
      </Selection>
      <Selection select_id="yellow_red_cards">
        <Descriptor>
          <Statement mimeType="text/plain">
            I want to see who is penalized
          </Statement>
        </Descriptor>
      </Selection>
      <Selection select_id="polls">
        <Descriptor>
          <Statement mimeType="text/plain">
            I want to vote for one of the players
          </Statement>
        </Descriptor>
      </Selection>
    </Choice>
    <Component>
      <Condition require="no_information" />
      <Resource mimeType="text/xml" ref="http://server/betting.php" />
    </Component>
    <Component>
      <Condition require="information␣tables" />
      <Resource mimeType="text/xml" ref="http://server/tables.php" />
    </Component>
    <Component>
      <Condition require="information␣player_information" />
      <Resource mimeType="text/xml" ref="http://server/player_information.php" />
    </Component>
    <Component>
      <Condition require="information␣injured_players" />
      <Resource mimeType="text/xml" ref="http://server/injured_players.php" />
    </Component>
    <Component>
      <Condition require="information␣yellow_red_cards" />
      <Resource mimeType="text/xml" ref="http://server/yellow_red_cards.php" />
    </Component>
    <Component>
      <Condition require="information␣polls" />
      <Resource mimeType="text/xml" ref="http://server/polls.php" />
    </Component>
      <Annotation target="#vote">
        <Descriptor>
          <Statement mimeType="text/plain">
            I vote for Peter Check for man of the match!
          </Statement>
        </Descriptor>
      </Annotation>
    </Item>
  </Container>
</DIDL>
```

Listing C.2: XML-feed example - *TV3.xml*

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE tv SYSTEM "xmltv.dtd">

<tv source-info-url="http://fredag.dagbladet.no/tv/" source-data-url="http://fredag.dagbladet.no/tv/" generator-info-name\
      ="XMLTV" generator-info-url="http://membled.com/work/apps/xmltv/">
  <channel id="tv3.no">
    <display-name>TV3</display-name>
  </channel>
  <programme start="20070922063000␣+0200" stop="20070922081500␣+0200" channel="tv3.no">
    <title lang="no">Barnetreern</title>
    <desc lang="no">Et variert utvalg tegnefilmer til glede for baade liten og stor. Alle programmene er paa norsk.</desc\
          >
    <episode-num system="all-seasons">41</episode-num>
  </programme>
  <programme start="20070922081500␣+0200" stop="20070922085500␣+0200" channel="tv3.no">
    <title lang="no">TV-shop</title>
  </programme>
  <programme start="20070922085500␣+0200" stop="20070922092500␣+0200" channel="tv3.no">
    <title lang="no">Heksen Sabrina</title>
    <title lang="en">Sabrina the Teenage Witch</title>
    <desc lang="no">Amerikansk ungdomsserie. Sabrina er en av kandidatene til en journaliststilling i lokalavisen. \
          Samtidig faar den unge heksen irriterende e-post fra en ukjent avsender. Hun skriver tilbake og ber om at e-\
          postene stoppes. Det Sabrina ikke vet er at e-posten kommer fra hennes onde tvillingsoester, Katrina. Katrina \
          blir rasende naar hun mottar klagen, og sender Sabrina et virus som forvandler henne til en idiot. Dermed \
          minsker Sabrinas sjanser til aa faa journalist-stillingen betraktelig...</desc>
    <episode-num system="all-seasons">127</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922092500␣+0200" stop="20070922101500␣+0200" channel="tv3.no">
    <title lang="no">Dawson's␣Creek</title>
␣␣␣␣<desc␣lang="no">Amerikansk␣dramaserie.␣Joey␣avsloerer␣en␣viktig␣avgjoerelse␣hun␣har␣tatt␣angaaende␣studiene.␣Pacey␣\
      blir␣med␣Joey␣paa␣et␣formelt␣selskap␣for␣elever␣som␣oensker␣aa␣gaa␣paa␣et␣prestisjefylt␣college.</desc>
␣␣␣␣<episode-num␣system="all-seasons">66</episode-num>
␣␣␣␣<previously-shown␣/>
␣␣</programme>
␣␣<programme␣start="20070922101500␣+0200"␣stop="20070922110500␣+0200"␣channel="tv3.no">
␣␣␣␣<title␣lang="no">Dawson's Creek</title>
    <desc lang="no">Amerikansk dramaserie. Joey greier ikke aa glemme synet av Gretchen og Dawson som kysset. Dawson paa \
          sin side, faar mange nye filmideer.</desc>
    <episode-num system="all-seasons">67</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922110500␣+0200" stop="20070922125500␣+0200" channel="tv3.no">
    <title lang="no">Rampete unger</title>
    <title lang="en">Bratty Babies</title>
    <desc lang="no">Kanadisk familiefilm fra 2001. Familieventyr om to separerte foreldre som kjemper en hard kamp om aa \
          holde de ukontrollerbare barna sine i sjakk. Kampen blir ikke lettere naar barna blir involvert med en gjeng \
          klossete tyver. Medvirkende:Harry Hamlin, Lisa Rinna, Frances Bay. Regi:Harvey Frost.</desc>
  </programme>
  <programme start="20070922125500␣+0200" stop="20070922132500␣+0200" channel="tv3.no">
    <title lang="no">Du er hva du spiser UK</title>
    <desc lang="no">Britisk livsstilsserie. Denne gangen forsoeker Gillian aa redde engelske barn fra deres medbragte \
          matpakker. Hun besoeker en skolegaard og avdekker at ryggsekkene der er fulle av usunn ferdigmat, sukker, fett \
          og E-numre i fleng. Programleder:Gillian McKeith.</desc>
    <episode-num system="all-seasons">38</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922132500␣+0200" stop="20070922135500␣+0200" channel="tv3.no">
    <title lang="no">Nanny</title>
    <desc lang="no">Amerikansk komiserie. Fran oensker aa adoptere barna i familien, men besteforeldrene paa morssiden \
          stikker kjepper i hjulene.</desc>
    <episode-num system="all-seasons">126</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922135500␣+0200" stop="20070922144500␣+0200" channel="tv3.no">
    <title lang="no">Dr. Phil</title>
    <desc lang="no">Amerikansk talkshow. Hvor etisk er du? De fleste vet aa gjoere det rette valget hvis de blir \
          konfrontert med et vanskelig dilemma, men hva skjer naar skillet mellom rett og galt nesten hviskes helt ut? \
          Programleder:Dr. Phillip C. McGraw.</desc>
    <episode-num system="all-seasons">806</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922144500␣+0200" stop="20070922154500␣+0200" channel="tv3.no">
    <title lang="no">Designerspirene USA</title>
    <title lang="en">Project Runway</title>
    <desc lang="no">Amerikansk realityserie. Tolv talentfulle og ivrige designerspirer faar sitt livs sjanse i denne \
          realityserien. Supermodellen Heidi Klum er programleder, og for seriens heldige vinner venter det moteshow \
          foran New Yorks moteelite og bilder i motemagasinet Elle. Vi foelger de haapefulle mens de gyver loes paa sine \
          utfordringer, og faar et innblikk i moteverdens bakside med krangling, baksnakking, taarer og slit. Etter hver \
          episode vil en av deltakerne faa droemmen sin knust, men til slutt vil vi kanskje faa se moteverdens nye store \
          designerstjerne! Deltakerne moetes i New York, hvor de faar sitt foerste oppdrag som designere. Oppgaven viser \
          seg aa vaere langt fra det de hadde ventet: De blir sluppet loes paa bakrommet til et supermarked og bedt om aa\
           lage klaer av det de finner der! Austin vet ikke hvordan han skal klare aa lage klaer av maisblader, mens \
          Mario er ferdig med sitt dusjforhengantrekk etter bare ti minutter. Programleder:Heidi Klum.</desc>
    <previously-shown />
  </programme>
  <programme start="20070922154500␣+0200" stop="20070922164000␣+0200" channel="tv3.no">
    <title lang="no">Top Model Norge</title>
    <desc lang="no">Norsk realityserie. Den spennende finalen naermer seg, og de tre jentene i New York tror de alle vil \
          staa likt ved siste utstemming. Men de faar seg en stygg overraskelse naar de faar beskjed om at det blir en \
          ekstrautstemming og at en av dem ikke faar vaere med paa den siste fotoshooten. Stemningen faller betraktelig, \
          og det er tungt for de to siste jentene aa komme seg gjennom siste fotoshoot etter sjokket. Men de gjoer en \
          glimrende jobb, og juryen faar en vanskelig oppgave naar det endelige valget skal gjoeres. Med en fantastisk \
          bakgrunn paa en takterrasse i New York skal det hele endelig avgjoeres: Hvem blir Norges nye toppmodell? \
          Programleder:Kathrine Soerland.</desc>
    <episode-num system="all-seasons">11</episode-num>
  </programme>
  <programme start="20070922164000␣+0200" stop="20070922174000␣+0200" channel="tv3.no">
    <title lang="no">Charterfeber - tur-retur Kreta</title>
    <desc lang="no">Norsk realityserie. Det er guttas siste kveld paa byen, og de forsoeker tappert aa sjekke damer. \
          Cheerleaderene drar paa baattur, mens pensjonistene tar en togtur for aa se naermere paa oeya. Svein faar \
          opplaering i salg, mens Hilde og Arne faar massasje og besoeker Kretas eldste by. De doeve jentene blir \
          invitert hjem til en lokal familie.</desc>
    <episode-num system="all-seasons">5</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922174000␣+0200" stop="20070922184000␣+0200" channel="tv3.no">
    <title lang="no">Den rette for Tor</title>
    <desc lang="no">Norsk realityserie. Tor Milde skal velge og vrake blant 12 kjaerlighetssyke damer. Med seg har Tor \
          tre av sine beste venner: Kaare Magnus Bergh, Tone Lise Skagefoss og Aslaug Tveiten Ally. I luksurioese og \
          ekstravagante omgivelser vil vennetroikaen gjoere sitt ytterste for aa bli best mulig kjent med damene. Gjennom\
           elleville tester og overraskende oppgaver skal vennepanelet avklare beilernes virkelige personlighet. Vennenes\
           raad blir viktige, men til syvende og sist er det Tor selv som maa staa for det avgjoerende valget. Og prisen \
          for aa vinne Tors forelskelse: En romantisk ferie for to, som kanskje blir starten paa en livslang reise - \
          forhaapentligvis. I dag tar Kaare Magnus damene med paa en ellevill dag paa Tusenfryd. Ikke alle vaager seg opp\
           i de spektakulaere karusellene, men naar Tor endelig dukker opp med en kjempepremie blir stemningen paa topp. \
          Senere paa dagen faar damene proeve seg som stand up-komikere, men forholdet damene imellom er i ferd med aa \
          slaa sprekker, og noen foeler de maa ta et oppgjoer.</desc>
    <episode-num system="all-seasons">2</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922184000␣+0200" stop="20070922200000␣+0200" channel="tv3.no">
    <title lang="no">Robinsonekspedisjonen 2007</title>
    <desc lang="no">Norsk realityserie. Utstyrt med svaert faa hjelpemidler kjemper deltakerne i ekspedisjonen om aa \
          overleve ved hjelp av det de finner. Christer Falck tester deltakernes fysiske og psykiske utholdenhet gjennom \
          nervepirrende og krevende konkurranser. Her er det om aa gjoere aa ha kontroll over sine omgivelser, ryker en \
          deltaker ut, er han eller hun ute for godt. Deltakerne har blitt delt i to lag, Nord og Soer, og fra foerste \
          stund er godene ujevnt fordelt mellom lagene. Lag Nord lever i overflod med baade ris og ild, mens lag Soer \
          sliter med sulten. Alliansene har allerede begynt aa ta form, men naar Kenneth tar til aa floerte med jentene \
          faller det ikke i god jord hos de andre gutta. Overraskelsen blir stor naar nok en jente ankommer oeya og \
          skaper baade interesse og konflikter blant de oevrige deltakerne.</desc>
    <episode-num system="all-seasons">1</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922200000␣+0200" stop="20070922213000␣+0200" channel="tv3.no">
    <title lang="no">Sangstjerner</title>
    <desc lang="no">Norsk underholdningsserie. I "Sangstjerner" skal norske kjendiser konkurrere om aa bli landets nye \
          kjendissanger. Naa skal blant annet sogneprest Einar Gelius, kunstner Marianne Aulie, dekkbaron Tommy Sharif, \
          tidligere Froeken Norge-vinner Hege Toerresdal og idrettsutoever og sydpolfarer Cato Zahl Pedersen faa sjansen \
          til aa vise at kjendiser uten nevneverdig sangerfaring toer aa synge paa en scene. I den kritiske juryen som \
          skal bedoemme kjendisenes sanginnsats sitter Anita Skorgan, aage "Glam" Sten Nilsen og Christian Ingebrigtsen. \
          Hvert program er bygget rundt et musikalsk tema, og det blir opp til publikum aa avgjoere hvilken kjendis som i\
           hvert program maa forlate konkurransen.</desc>
    <episode-num system="all-seasons">3</episode-num>
    <previously-shown />
  </programme>
  <programme start="20070922213000␣+0200" stop="20070922235500␣+0200" channel="tv3.no">
    <title lang="no">Fallen</title>
    <desc lang="no">Amerikansk groesser fra 1998. Drapsmannen Edgar Reese har faatt doedsdom og hans dager i fengsel er \
          naa talte. Paa vei til gasskammeret synger han, mens politietterforsker John Hobbes drar et lettelsens sukk. \
          Noen dager senere begynner bekjente aa synge samme sangen som Reese sang i gasskammeret, og Hobbes faar hoere \
          at det kanskje er den falne engelen Azazel som staar bak. Azazel er doemt til aa flakke rundt paa jorden uten \
          form, og tar bolig hos hvem han vil ved naerkontakt. Hobbes blir tvunget til aa drepe dem som er besatt av \
          Azazel og maa baade renvaske sitt navn og beskytte sin familie. Medvirkende:Denzel Washington, John Goodman, \
          Donald Sutherland, Embeth Davidtz, James Gandolfini. Regi:Gregory Hoblit. (15. aar)</desc>
  </programme>
  <programme start="20070922235500␣+0200" stop="20070923021000␣+0200" channel="tv3.no">
    <title lang="no">De-Lovely</title>
    <desc lang="no">Amerikansk-britisk musikkdrama fra 2004. Et musikalsk portrett av den amerikanske komponisten Cole \
          Porter. Vi ser tilbake paa hans liv og virke som var like fargerike som hans sceneshow. Handlingen utspiller \
          seg paa scenen der livet hans passerer i revy. Porters samtid og vennekrets stiger frem i scenelyset - og vi \
          faar ogsaa kjennskap til hans dype og kompliserte kjaerlighetsforhold til sin kone og muse, Linda Lee Porter. \
          Medvirkende:Kevin Kline, Ashley Judd, Jonathan Pryce, Kevin McNally, Kevin McKidd. Regi:Irwin Winkler.</desc>
  </programme>
</tv>
```

Listing C.3: Code for XMLguide - *Mainclass.java*

```
package xmlguide;




import javax.microedition.lcdui.*;
import javax.microedition.midlet.MIDlet;


public class Mainclass extends MIDlet{

  private GuideCanvas MyMap;//our main canvas

  public  Mainclass(){
    System.out.println("Start");
    MyMap = new GuideCanvas(this);

  }

  protected void startApp(){
    Display.getDisplay(this).setCurrent(MyMap);
    try{
      Thread myThread = new Thread(MyMap);
      myThread.start();
      System.out.println("Go␣thread!");
    }catch (Error e) {
          destroyApp(false);
          notifyDestroyed();
          System.out.println("failed␣to␣start␣thread");
    }
  }



  public void destroyApp(boolean eh){
  }

  protected void pauseApp() {
    }
}
```

Listing C.4: Code for XMLguide - *GuideCanvas.java*

```java
package xmlguide;


import javax.microedition.lcdui.*;
//import javax.microedition.lcdui.game.*;
import javax.microedition.midlet.MIDlet;
import java.util.*;


public class GuideCanvas extends Canvas implements Runnable, CommandListener{

  private MIDlet MotherMid;

  //button for exiting application
  private Command cmd_exit = new Command("Avslutt",Command.EXIT,1);

  Image imgbuffer;

  ProgramClass progShowinInfo;

  //channel list
  Vector channels = new Vector();

  public GuideCanvas(MIDlet mother){
    //set mother midlet, so that commands here can kill the app. I'm a crappy porgrammer
    MotherMid = mother;
    st.updateTime();
    //set up buttons
    addCommand(cmd_exit);
    setCommandListener(this);
    setFullScreenMode(true);//set fullscreen

    if(true){
    //init XML
    XmlGuideParser.initXML(this);
    }else{
    channels.addElement(new ChannelClass("TV␣Norge",  "tvn",    "/tv-tvn.png",    this));
    channels.addElement(new ChannelClass("NRK",     "nrk",    "/tv-nrk1.png",   this));
    channels.addElement(new ChannelClass("NEPEKANALEN", "nep",    "/tv-tv3.png",    this));


    //add some programs
    ProgramClass p;
    p = new ProgramClass("Ut␣i␣naturen", "20070625190000", "20070625193000",  "Vi␣utforsker␣villmark␣og␣dyr␣og␣saann␣med␣\
          Arvid␣Hippie");
    getChannel(0).addProgram(p);
    p = new ProgramClass("Mat␣med␣Maans", "20070625193000", "20070625210000", "I␣kveld:␣En␣kulinarisk␣reise␣til␣Skaane!")\
          ;
    getChannel(0).addProgram(p);
    p = new ProgramClass("Slankekrigen", "20070626213000", "20070626223000",  "Grethe␣Rhode␣treffer␣Dr.␣Fedon␣Lindberg,␣\
          dette␣blir␣fett!");
    getChannel(0).addProgram(p);

    p = new ProgramClass("Nyhetene", "20070625191500", "20070625194500",    "Nyhetene,␣forst␣paa␣norsk,␣saa␣paa␣samisk␣og\
          ␣saa␣paa␣tegnspraak");
    getChannel(1).addProgram(p);
    p = new ProgramClass("Vaeret", "20070625195000", "2007062520000",     "I␣Trondheim:␣Her␣skjer␣mye␣goy!");
    getChannel(1).addProgram(p);

    p = new ProgramClass("Nepetid!", "20070625195000", "20070625203000",    "Vi␣besoker␣Nepegaarden␣til␣Thorgeir␣og␣Beate\
          ,␣og␣saa␣lager␣vi␣Nepesuppe.");
    getChannel(2).addProgram(p);
    p = new ProgramClass("Show␣om␣Neper", "20070625210500", "20070625220000", "Episode␣18:␣Nepalske␣neper␣smaker␣ekstra␣\
          godt␣i␣nepesuppen,␣Jan␣Brimi␣kommer␣som␣gjest.");
    getChannel(2).addProgram(p);
    }
    for(int i = 0 ; i< channels.size(); i++)
      System.out.println("kanal:␣" + getChannel(i).name);

    //set current channel and program
    selectChannel(0);
    selectProgram( findProgramClosestTo(st.getNow(),0) );
    progShowinInfo=null;



    while(st.sw==0 && st.sh ==0){//loop till we get dimentions
      //preform a repaint, so that we get the dimentions right
      repaint();
      //set dimentions
      st.sw = getWidth();
      st.sh = getHeight();
    }
    //if the screen is higher than the size of the channels
    //st.sh = Math.min(st.channelSegmentHeight * channels.size(), st.sh);
    //create buffer
```

```
      imgbuffer = Image.createImage(st.sw, st.sh);
  }

  public ChannelClass getChannel(int index){return (ChannelClass)channels.elementAt(index);}
  public ChannelClass getChannel(String id){
    for(int i=0; i<channels.size(); i++)
      if(getChannel(i).getID().equals(id))return getChannel(i);
    return null;
  }
  public void addChannel(ChannelClass ch){channels.addElement(ch);}

  public void commandAction(Command c,Displayable d){
    if(c==cmd_exit){
      MotherMid.notifyDestroyed();
    }
  }


  int targetOffsetX;

  public void run(){
    int i=0;
    while(true){
      synchronized(this) {
        try {wait(10);}
        catch(Exception e){}
      }
      //move towards target
      if(Math.abs(st.offsetx-targetOffsetX) > 6){
        int diff = Math.abs(st.offsetx-targetOffsetX);
        //st.offsetSpeed = (int)(Math.sqrt(diff));
        st.offsetSpeed = (int)(diff/10);
        if(st.offsetSpeed>10)st.offsetSpeed=10;
        if(st.offsetSpeed<1)st.offsetSpeed=1;

        if(st.offsetx > targetOffsetX) st.offsetx -=st.offsetSpeed;
        if(st.offsetx < targetOffsetX) st.offsetx +=st.offsetSpeed;
        //System.out.println(st.offsetSpeed);
        //System.out.println("offset: " + st.offsetx +"/" + targetOffsetX  );
        this.repaint();
      }

      //System.out.println("hee");
      if(i==0)st.updateTime();
      i++;
      i%=40;

    }
  }

  protected void keyPressed(int keyCode){
    //this sub looks for navigation commands
    int action = getGameAction(keyCode);
    if(action==LEFT )selectProgram(st.selectedProgram-1);
    if(action==RIGHT)selectProgram(st.selectedProgram+1);
    if(action==DOWN) selectChannel(st.selectedChannel+1);
    if(action==UP)   selectChannel(st.selectedChannel-1);

    if(action==FIRE) {
      if(progShowinInfo == null)
        progShowinInfo = getChannel(st.selectedChannel).getProgram(st.selectedProgram);
      else
        progShowinInfo = null;
    }
    if(keyCode==KEY_POUND){st.secsPerPixel+=10;updateTargetOffset(); st.offsetx = targetOffsetX;}
    if(keyCode==KEY_STAR ){st.secsPerPixel-=10;updateTargetOffset(); st.offsetx = targetOffsetX;}

    //if(keyCode==KEY_NUM0) ShowCommerce = !ShowCommerce;

    //System.out.println("key: " + keyCode);
    repaint();
  }
  int findProgramClosestTo(long targettime, int chanID){
    long bestdiff = Long.MAX_VALUE;
    int bestpos = 0, i=0;
    ProgramClass p;
    while( (p=getChannel(chanID).getProgram(i)) != null ){
      long diff = Math.abs(p.getStart() - targettime);
      if(diff<bestdiff){
        bestdiff=diff; bestpos=i;
      }
      i++;
    }
    return bestpos;
  }

  public void selectChannel(int chanID){
    //validate chanID
    if(chanID<0 || chanID>=channels.size())return;
    long targettime = getChannel(st.selectedChannel).getProgram(st.selectedProgram).getStart();
    st.selectedChannel = chanID;
    //find the program closest to current
    int bestpos = findProgramClosestTo(targettime, chanID);
    selectProgram(bestpos);
  }
  public void selectProgram(int progID){
    //validate progID
    ProgramClass p = getChannel(st.selectedChannel).getProgram(progID);
    if(p == null) return;

    st.selectedProgram = progID;
    //center screen on program
    updateTargetOffset();
  }
  public void updateTargetOffset(){
    ProgramClass p = getChannel(st.selectedChannel).getProgram(st.selectedProgram);
    if(p == null) return;
    targetOffsetX = st.channelLeftMargin +st.nowMargin -st.dateToPosition(p.getStart(),true);
  }
  protected void paint(Graphics targetGraphics){
    int x;
    int sw=st.sw;
    int sh=st.sh;
    //System.out.println("REDRAW");

    Graphics g = imgbuffer.getGraphics();


    long time = System.currentTimeMillis();

    //break out and draw fullscreen info
    if(progShowinInfo!=null) {
      progShowinInfo.drawFullscreenInfo(g);
      targetGraphics.drawImage(imgbuffer, 0,0, 0);
      return;
    }

    //clear background
    g.setColor(st.colBackground);
    g.fillRect(0,0,sw,sh);
    g.setColor(0,0,0);


    //draw all channels
    for(int i = 0 ; i< channels.size(); i++){
      getChannel(i).Draw(g, st.channelSegmentHeight*i, st.channelSegmentHeight, sw);
    }

    int heightOfChannels =  st.channelSegmentHeight * channels.size();

    //draw each upcoming hour
    g.setColor(255,255,255);
    long starttime = st.positionToDate(0,false);
    long endtime = st.positionToDate(st.sw,false);
    //round off to closest hour
    starttime = (long)(starttime/(1000*60*60))*1000*60*60;
    for(long t=starttime;t<endtime; t+=1000*60*60){
      x = st.dateToPosition(t,false);
      g.drawLine(x, 0, x, heightOfChannels);
      g.drawString(st.UnitimeToClock(t).substring(0,2), x, heightOfChannels+1, 0);
    }
    g.setColor(0xffffff);
    g.fillRect(0, sh-20, 0, sh);
    g.setColor(0);


    //draw CURRENT time indicator
    x = st.dateToPosition(st.getNow(),false);
    g.setColor(255,0,0);
    g.drawLine(x,   0, x,  heightOfChannels);
    g.drawLine(x+1, 0, x+1, heightOfChannels);


    //draw background for channel logos
    g.setColor(st.colLeftMargin);
    g.fillRect(0, 0, st.channelLeftMargin, sh);
    g.setColor(st.colLeftMarginFrame);
    g.drawLine(st.channelLeftMargin, 0, st.channelLeftMargin, sh);

    //draw all channelnames
    g.setColor(st.colChannelNames);
    for(int i = 0 ; i< channels.size(); i++){
      //draw name
      ChannelClass c = getChannel(i);
      if(c.getLogo() != null )
        g.drawImage(c.getLogo(), 2, i *st.channelSegmentHeight + 3, 0);
      else{
        g.drawString(c.getName().substring(0,Math.min(4,c.getName().length())), 2, i *st.channelSegmentHeight + 3, 0);
      }
    }
    //draw current date
    g.setColor(st.colLeftMargin);
    g.fillRect(0, sh-20, sw, sh);
    g.setColor(st.colLeftMarginFrame);
    g.drawLine(st.channelLeftMargin, sh-20, sw, sh-20);
    g.setColor(0);
    starttime = st.positionToDate(0,false);
    g.drawString( st.UnitimeToStringDate(starttime) , st.channelLeftMargin+2, sh-18, 0);


    //flip buffer
    targetGraphics.drawImage(imgbuffer, 0,0, 0);

    //System.out.println("time: " + (System.currentTimeMillis() - time));
  }



}
```

Listing C.5: Code for XMLguide - *XmlGuideParser.java*

```java
package xmlguide;

import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

import org.kxml.Xml;
import org.kxml.io.*;
import org.kxml.parser.ParseEvent;
import org.kxml.parser.XmlParser;


public class XmlGuideParser {
  static public void initXML(GuideCanvas ownerGuide){
    int i=0;
    //xml stuff
    XmlParser parser = null;
    //Document doc = new Document();

    String resfile_name = "/mine4.xml";
    System.out.println("parsing␣XML");
    try {
      // Read in the resource
      InputStream in = ownerGuide.getClass().getResourceAsStream(resfile_name);
      // Turn it into a reader
      InputStreamReader isr = new InputStreamReader(in);
      // initialize the parser with it.
      parser = new XmlParser(isr);

      // Pass the parser to the document.  At this point the
      // entire resource is parsed and now resides in memory.
      //doc.parse( parser );

      ParseEvent event = null;
      while((event = parser.read()).getType() != Xml.END_DOCUMENT ){
        if (event.getType() == Xml.START_TAG) {
          String eventName = event.getName();
          //Parse channels -------------------------------------------------------------------
          if(eventName.equals("channel")) {
            String chanID = event.getAttribute("id").getValue();
            String chanName = "";
            while((event = parser.read()).getType() != Xml.END_DOCUMENT  ){
              String name = event.getName();
              int type = event.getType();
              //break if we are at the end of the channel tag
              if(type == Xml.END_TAG && name.equals("channel")) break;
              //if it's not a text tag; continue
              if(type != Xml.TEXT)continue;
              chanName = event.getText();
            }
            System.out.println("New␣Channel:␣" + chanName + "␣" + chanID);
            ownerGuide.addChannel(new ChannelClass(chanName,chanID,chanID + ".png",ownerGuide));
          }
          //Parse programs -------------------------------------------------------------------
          if(eventName.equals("programme")){
            String progStart = event.getAttribute("start").getValue();
            String progEnd   = event.getAttribute("stop" ).getValue();
            String progChan  = event.getAttribute("channel").getValue();
            String progDesc = "";
            String progTitle = "";
            //get channel. since there is no point continuing if it is NULL
            ChannelClass ch = ownerGuide.getChannel(progChan);

            boolean settingDesc=false, settingTitle=false;
            while((event = parser.read()).getType() != Xml.END_DOCUMENT  && ch!=null){
              String name = event.getName();
              int type = event.getType();
              //break if we are at the end of the program tag
              if(type == Xml.END_TAG && name.equals("programme")) break;
              //sett mode
              if(type == Xml.END_TAG && name.equals("desc")) settingDesc = false;
              if(type == Xml.END_TAG && name.equals("title")) settingTitle = false;
              if(type == Xml.START_TAG && name.equals("desc")) settingDesc = true;
              if(type == Xml.START_TAG && name.equals("title")) settingTitle = true;

              //if it's not a text tag; continue
              if(type != Xml.TEXT)continue;
              if(settingTitle)progTitle = event.getText();
              if(settingDesc) progDesc = event.getText();
            }


            //System.out.println(progStart + " - " + progEnd+ " on " + progChan);

            if(ch!=null){
              ch.addProgram(new ProgramClass(progTitle,progStart,progEnd,progDesc));
              i++;
              if(i%20==0)
                System.out.println("loaded␣" + i + "␣program␣blocks");
            }
          }
        }
        //if (i>20) break;
      }

      parser = null;
      System.out.println("done␣parsing");
    } catch (IOException ioe) {
      // report error
      System.err.println("XML␣Parsing␣Error:␣" + ioe);
      ioe.printStackTrace();

      parser = null;
      //doc = null;

      return;
    }catch (Exception e) {
      System.out.println(e);
      return;
    }

    /*
    //show some data
    Element root = doc.getRootElement();
    for(int i=0; i <root.getChildCount();i++){
      Element n = root.getElement(i);
      if(n== null)continue;
      //a new program entry
      if(n.getName().equals("programme")){
        //found a new progam
        String progName="";
        String progStart="";
        String progEnd="";
        String progChannel="";
        String progDesc="";
        //get attributes
        progStart = n.getAttribute("start").getValue();
        progEnd = n.getAttribute("stop").getValue();
        progChannel = n.getAttribute("channel").getValue();

        //go through every piece of info in this program block
        for(int j=0;j<n.getChildCount();j++){
          Element info = n.getElement(j);
          if(info==null)continue;
          if(info.getName().equals("title")) progName = info.getText();
          if(info.getName().equals("desc")) progDesc = info.getText();
        }
        System.out.println("PROG:" + progName + " " + progStart + " to " + progEnd);
        ProgramClass p = new ProgramClass(progName, progStart, progEnd, progDesc);
        ChannelClass c = getChannel(progChannel);
        if(c!=null)c.addProgram(p);
      }
      //a new channel entry
      if(n.getName().equals("channel")){
        String chanID = n.getAttribute("id").getValue();
        String chanName="";
        for(int j=0;j<n.getChildCount();j++){
          Element info = n.getElement(j);
          if(info==null)continue;
          if(info.getName().equals("display-name")) chanName = info.getText();
        }
        System.out.println("CHAN:" + chanName + " : " + chanID) ;
        channels.addElement(new ChannelClass(chanName,chanID,chanID + ".png",this));
      }
    }*/
  }
  static private void parseXmlChannel(XmlParser parser, GuideCanvas guide){
    ParseEvent event = null;
    try{
      while ((event = parser.peek()).getType() != Xml.END_DOCUMENT) {
        //return if we find the end of this tag
        if (event.getType() == Xml.END_TAG && event.getName().equals("channel")) return;
        System.out.println(event.getAttribute("id"));
        event = parser.read();



      }
    }catch(Exception e){}
  }

}
```

Listing C.6: Code for XMLguide - *ChannelClass.java*

```java
package xmlguide;

import java.util.Vector;

import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;

public class ChannelClass {

  String name;
  String ID;
  public Vector programs = new Vector();
  GuideCanvas owner;

  private Image imgLogo;

  public ChannelClass(String name,String ID, String strLogo, GuideCanvas owner){
    this.owner = owner;
    this.name = name;
    this.ID = ID;
    //load image
    try{
      imgLogo = Image.createImage("/" + strLogo);
    } catch (java.io.IOException e) {
      System.out.println("failed␣to␣load␣image:" + strLogo);
    }
  }

  public ProgramClass getProgram(int index){
    if(index<0 || index>=programs.size()) return null;
    return (ProgramClass)programs.elementAt(index);
  }

  public void addProgram(ProgramClass p){
    //check to not add too short programs, or programs that start before they end
    if(p.getEnd()- p.getStart() < st.shortestProgTime) return;
    p.setOwner(this);
    programs.addElement(p);
  }

  String fitStringToWidth(String s,int pixwidth){
    int end = Math.min(s.length(), pixwidth/10 +1);
    return s.substring(0,end);
  }

  public String getName(){return name;}
  public Image getLogo(){return imgLogo;}
  public String getID(){return ID;}

  public void Draw(Graphics g, int starty, int height, int width){
    g.setColor(255,255,255);
    //draw horizontal divider
    g.drawLine(st.channelLeftMargin+1, starty+height, width, starty+height);

    //draw program entries
    for(int i=0; i<programs.size(); i++){
      ProgramClass p = getProgram(i);
      int sx = st.dateToPosition(p.getStart(),false);
      int ex = st.dateToPosition(p.getEnd(),false);
      if(sx>st.sw || ex < 0) continue; //don't do ANYTHING for programs outside the screen

      g.setColor(150, 150,150);
      //if this is the selected program and channel, hightligt
      if(owner.getChannel(st.selectedChannel)== this &&  i == st.selectedProgram) g.setColor(0, 255,0);


      //System.out.println(sx + " to " + ex);
      //background
      g.fillRect(sx, starty+1, ex-sx, height-2);
      //text
      g.setColor(0, 0,0);
      g.drawString(fitStringToWidth(p.getName(),ex-sx),sx+1,starty+3,0);
      //frame
      g.setColor(100, 100,100);
      g.drawRect(sx, starty+1, ex-sx, height-2);

    }
  }
}
```

Listing C.7: Code for XMLguide - *ProgramClass.java*

```java
package xmlguide;

import java.util.Calendar;
import java.util.Vector;

import javax.microedition.lcdui.*;


public class ProgramClass {
  long start;
  long end;
  ChannelClass ownerchannel;

  String name;
  String description;

  public ProgramClass(String name, String start, String end, String description){
    this.name  = name;
    this.description = description;
    setEnd(end);
    setStart(start);
  }

  public void drawFullscreenInfo(Graphics g){
    //TODO:GOD I HATE OOP!
    //clear background
    g.setColor(200,200,200);
    g.fillRect(0,0,st.sw,st.sh);
    g.setColor(0,0,0);

    //draw program name
    g.drawString(name, 3,3,0);
    //draw time and channel
    g.setColor(80,80,80);
    g.drawString( st.UnitimeToClock(start)+"␣-␣" + st.UnitimeToClock(end) + "␣paa␣" + ownerchannel.getName(), 5,  15, 0);


    //draw description
    if(st.sw < 5 )return;//too small screen
    g.setColor(0,0,0);
    Vector lines = st.splitByWidth(description, st.sw-4, g.getFont());
    for(int i=0; i< lines.size(); i++)
      g.drawString(lines.elementAt(i).toString().trim(), 5, 27+ 12*i, 0);
  }

  public void setOwner(ChannelClass ow){ownerchannel=ow;}
  public ChannelClass getOwner(){return ownerchannel;}

  //20070611060000 = 2007 06 11 0600
  public void setStart(String str){
    start = st.stringToUnitime(str);
  }
  public void setEnd(String str){
    end = st.stringToUnitime(str);
  }

  public long getStart(){
    //return (new Date().getTime() + (1000*60*60)); // one hour in the future
    return start;
  }

  public long getEnd(){
    //return (new Date().getTime() + (1000*60*90)); // one hour and 30 min in the future
    return end;
  }
  public String getName(){return name;}
  public String getDesc(){return description;}



}
```

Listing C.8: Code for XMLguide - *st.java*

```java
package xmlguide;

import java.util.Calendar;
import java.util.Date;
import java.util.Vector;

import javax.microedition.lcdui.Font;

public class st {



  static public int sw;
  static public int sh;

  static public int offsetx;
  static public float offsetSpeed;

  static public int secsPerPixel = 90;

  //programs shorter than this will not be shown
  public static long shortestProgTime = 1000*60;

  public static int selectedChannel;
  public static int selectedProgram;

  //height of one channel block
  public static int channelSegmentHeight = 25;

  //the size of the left margin, the background for the channel logos
  public static int channelLeftMargin = 25;

  //how many pixels of "the past" should we show?
  public static int nowMargin = 25;


  public static int colLeftMargin     = 0x0587A7;
  public static int colLeftMarginFrame  = 0x0A6272;

  public static int colChannelNames   = 0xFFFFFF;

  public static int colBackground     = 0x323296;

  public static String[] weekdays = {"Mandag","Tirsdag","Onsdag","Torsdag","Fredag","Lordag","Sondag"};
  public static String[] months = {"Jan","Feb","Mar","Apr","Mai","Jun","Jul","Aug","Sep","Nov","Des"};

//  returns the hour:min format of a unitime Long
  static public String UnitimeToClock(long time){
    Calendar c = Calendar.getInstance();
    c.setTime(new Date(time));
    if(c.get(Calendar.MINUTE)<10)return c.get(Calendar.HOUR_OF_DAY) + ":0" + c.get(Calendar.MINUTE) ;
    return c.get(Calendar.HOUR_OF_DAY) + ":" + c.get(Calendar.MINUTE) ;
  }

  static public String UnitimeToStringDate(long time){
    Calendar c = Calendar.getInstance();
    c.setTime(new Date(time));
    int weekday = (c.get(Calendar.DAY_OF_WEEK) +5)%7;
    return weekdays[weekday] + "␣"
    + Integer.toString(c.get(Calendar.DAY_OF_MONTH)) + ".␣"
    + months[c.get(Calendar.MONTH)];
  }

  static long nowtime;
  //returns the current time in unitime
  static public long getNow(){return nowtime;}

  static public void updateTime(){
    nowtime = System.currentTimeMillis() + 1000 * 60 * 60 * 0;//(new Date()).getTime();
    //System.out.println("TIME UPDATE: ("  +  nowtime + ")"  + UnitimeToStringDate(nowtime));
    //nowtime = stringToUnitime("20070625183100");
  }

  static public int dateToPosition(long timepoint, boolean absolute){
    //one pixel = 90 seconds
    int scale = 1000 * secsPerPixel;
    //now is 0 pixels to the right
    long now = getNow();

    //System.out.println("from " + now + " to " + timepoint);

    long retval = timepoint - now;
    retval = retval/scale;
    if(!absolute) retval += offsetx;
    return (int)retval + channelLeftMargin + nowMargin ;
  }
  static public long positionToDate(int x, boolean absolute){
    //one pixel = 90 seconds
    int scale = 1000 * secsPerPixel;

    long retval = x - channelLeftMargin - nowMargin ;
    if(!absolute) retval -= offsetx;
    retval *= scale;

    long now = getNow();

    retval = retval + now;

    return retval;
  }

  static public long stringToUnitime(String str){
    str = str.trim();
    int year  = Integer.parseInt( str.substring(0,4));
    int month = Integer.parseInt(str.substring(4,6));
    int day   = Integer.parseInt(str.substring(6,8));
    int hour  = Integer.parseInt(str.substring(8,10));
    int min   = Integer.parseInt(str.substring(10,12));
    // 20070625075400 +0200
    //int gmtoffset=0;
    //if(str.endsWith("+0100"))gmtoffset = 1;
    //if(str.endsWith("+0200"))gmtoffset = 2;


    Calendar c = Calendar.getInstance();
    c.set(Calendar.YEAR , year);
    c.set(Calendar.MONTH , month-1);
    c.set(Calendar.DAY_OF_MONTH , day);
    c.set(Calendar.HOUR_OF_DAY  , hour);
    c.set(Calendar.MINUTE, min);
    return c.getTime().getTime() ;//+ gmtoffset * 1000 * 60 * 60 ;
  }

//   Replace function
  public static String replace(String _text, String _searchStr, String _replacementStr) {
    // String buffer to store str
    StringBuffer sb = new StringBuffer();

    // Search for search
    int pos = _text.indexOf(_searchStr);

    // Iterate to add string
    while (pos != -1)
    {
      sb.append(_text.substring(0, pos)).append(_replacementStr);

      _text = _text.substring(pos + _searchStr.length());
      pos = _text.indexOf(_searchStr);
    }

    // Create string
    sb.append(_text);

    return sb.toString();
  }

  public static Vector splitByWidth(String original, int width, Font f){
    Vector ret = new Vector();
    while(original.length() >0){
      int i = 0;
      String temp ="";
      String line ="";
      while(f.stringWidth(line.trim() + temp.trim()) <= width){
        temp="";
        while(true){
          temp = temp + original.substring(i,i+1);
          i++;
          //break if we hit the end
          if(i==original.length())break;
          //break if we hit a space
          if(original.substring(i,i+1).equals("␣"))break;
        }
        line = line + temp;
        //System.out.println("" + temp.trim() + " " + f.stringWidth(line + temp));
        //break if we hit the end
        if(i==original.length())break;

      }
      ret.addElement(line);
      original = replace(original,line  ,"");
    }
    return ret;
  }
}
```