

Juha-Matti Lehtinen

Context-Aware DVB-H Service Software Framework

Faculty of Information and Natural Sciences

Master's Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology in Espoo 31.01.2011.

Supervisor:

Prof. Petri Vuorimaa

Instructor:

D.Sc. (Tech.) Timo Laakko



Author: Juha-Matti Lehtinen

Title: Context-Aware DVB-H Service Software Framework

Date: 31.01.2011

Language: English

Number of pages: 7 + 81

Faculty of Information and Natural Sciences

Department of Media Technology

Professorship: Media Technologies

Code: T-111

Supervisor: Prof. Petri Vuorimaa

Instructor: D.Sc. (Tech.) Timo Laakko

Context is any information that can be used to characterize the situation of an entity. Context-aware system uses this context to provide relevant information and/or services to user. *Context-awareness* is a concept, which is increasingly both prevalent and essential in mobile services and applications. Additionally, *personalization* is a relevant concept in the field of mobile services. It is used to match the offered content and services to user's personal preferences.

Mobile service providers require affordable content delivery methods. One viable alternative is *DVB-H*, a digital TV specification, which is specifically designed for mobile environments. Utilizing broadcasting, DVB-H is able to serve large amounts of customers in extremely cost-efficient way. Combining context-awareness with DVB-H is potentially a remarkable scenario: users of this kind of service are able to consume content specific to their needs and preferences, while service providers gain maximal profits due to the inexpensive DVB-H delivery network.

Based on this scenario, this Thesis presents i) a context-aware DVB-H service software framework that was developed and ii) a reference implementation, which was actualized to validate the framework design. The fundamental research question was: *in what way can context-awareness and personalization be utilized in recommending DVB-H-mediated content to users*. The developed framework was designed based on the requirements derived from earlier work related to context-aware system architectures and on guidelines related to designing context-aware mobile services and applications. DVB-H-specific context-awareness problems were considered to complete the design.

The functioning of the reference implementation was confirmed with test scenarios, which validate the set requirements and provide valuable insights for the future developers of Context-Aware DVB-H Service. The scenarios validate that it is possible and even beneficial to create a context-aware mobile recommendation service, whose target content delivery medium is DVB-H.

Keywords: context, context-awareness, personalization, DVB-H, mobile service

Tekijä: Juha-Matti Lehtinen		
Työn nimi: Context-Aware DVB-H Service Software Framework		
Päivämäärä: 31.01.2011	Kieli: Englanti	Sivumäärä: 7 + 81
Informaatio- ja luonnontieteiden tiedekunta		
Mediatekniikan laitos		
Professori: Mediatekniikat	Koodi: T-111	
Valvoja: Prof. Petri Vuorimaa		
Ohjaaja: TkT Timo Laakko		
<p>Kontekstilla tarkoitetaan tietoa, jota voidaan käyttää jonkin kokonaisuuden tilanteen luonnehtimiseen. Kontekstittietoinen järjestelmä hyödyntää tätä kontekstia tuottaakseen merkityksellistä tietoa ja/tai palveluita järjestelmän käyttäjälle. Kontekstittietoisuus on yhä olennaisempi käsite tulevaisuuden mobiilipalveluissa ja –sovelluksissa. Personointi on myös olennainen käsite: sitä käytetään tarjotun sisällön ja käyttäjän kiinnostusten väliseen vertailuun.</p> <p>Mobiilipalveluntarjoajat tarvitsevat edullisia sisällönlähetystapoja. Eräs mahdollinen ratkaisu tähän on DVB-H-tekniikan käyttö. DVB-H on digitaal-TV-spesifikaatio, joka on suunniteltu käytettäväksi mobiiliympäristöissä. DVB-H:n avulla voidaan tavoittaa suuri joukko asiakkaita erittäin kustannustehokkaasti. Kontekstittietoisuuden ja DVB-H-tekniikan yhdistäminen mobiilipalveluksi on potentiaalisesti merkittävä skenaario: tämän kaltaisen palvelun käyttäjät pystyisivät kuluttamaan juuri sellaista sisältöä, jota he tarvitsevat ja josta he ovat kiinnostuneita; toisaalta palveluntarjoajat hyötyvät huomattavasti tästä rahallisesti.</p> <p>Edellä kuvattuun skenaarioon perustuen tässä diplomityössä esitetään i) kehitetty kontekstittietoisuuden DVB-H-palvelun ohjelmistokehitys ja ii) referenssitoteutus, joka tehtiin ohjelmistokehityksen suunnittelun validoimiseksi. Tutkimuskysymys oli seuraavanlainen: <i>millä tavoin kontekstittietoisuutta ja personointia voidaan hyödyntää DVB-H:n avulla välitetyn sisällön suosittelumiseksi käyttäjille</i>. Kehitetty ohjelmistokehitys suunniteltiin aiempaan tutkimukseen ja kontekstittietoisuuden mobiilipalveluiden ja -sovelluksien kehittämissuosituksiin perustuviin vaatimuksiin nojautuen. Ongelmat, jotka liittyvät kontekstittietoisuuden hyödyntämiseen DVB-H:ssa, käsiteltiin myös ohjelmistokehityksen kehittämisen täydentämiseksi.</p> <p>Referenssitoteutuksen vaatimusten mukainen toiminta validoitiin testiskenaarioilla. Skenaarioista saatujen tulosten avulla kontekstittietoisuuden DVB-H-palvelun kehittäjä voi edelleen parantaa ohjelmistokehityksen toimintaa. Skenaariot osoittavat, että kontekstittietoisuuden mobiilisuosittelevan palvelun kehittäminen DVB-H:lle ei ole vain mahdollista mutta myös suotavaa.</p>		
Avainsanat: konteksti, personointi, kontekstittietoisuus, DVB-H, mobiilipalvelu		

Preface

Work on this Thesis has been carried out from May 2010 to January 2011 as part of the CoDe project at VTT Technical Research Centre of Finland.

The work done in this Thesis has personally been an educative and interesting challenge. I was given an independent opportunity to be creative and innovative in developing and actualizing a new kind of mobile service. Personally, I felt I was creating something novel, and in some tangible way contributing to the mobile media research society. I would like to extend my gratitude to VTT for giving me the opportunity to improve my insight into the world of academic research.

I would like to thank Mr. Timo Laakko, the project leader of the CoDe project, for the smooth and fluent guidance throughout the writing process. The tips related to both writing this Thesis and designing the practical work were always generic enough, which promoted my own self-learning skills and the ability to solve critically and independently problems related to academic work. Equally, I would like to thank Mr. Petri Vuorimaa for the communicative and wise Thesis supervision, and for pointing out the necessary changes needed to render this Thesis to its finalized form.

The year 2010 has been a year for personal growth to me in many ways. In the midst of all of this, my dear Family and my dear Friends have always been there for me; I feel privileged to know such a variety of wonderful people, supporting me in my endless endeavors. I thank you all for being an essential part of my life. The completion of this Thesis would not have been possible without your energizing presence.

Otaniemi, 31.01.2011

Juha-Matti J. Lehtinen

TABLE OF CONTENTS

ABSTRACT	II
TIIVISTELMÄ	III
PREFACE	IV
TABLE OF CONTENTS	V
ABBREVIATIONS	VII
1 INTRODUCTION	1
1.1 BACKGROUND AND MOTIVATION	1
1.2 RESEARCH QUESTIONS AND GOALS	2
1.3 ORGANIZATION OF THE THESIS	3
2 BACKGROUND	4
2.1 DVB-H TECHNOLOGY	4
2.2 CONTEXT AND ITS TYPES	7
2.3 CONTEXT REPRESENTATION MODELS IN MOBILE SYSTEMS	8
2.4 CONTEXT-AWARENESS: INFERRING SITUATION FROM SENSOR DATA	9
2.5 PERSONALIZATION AND ADAPTATION OF CONTENT	10
2.6 EARLIER WORK	12
2.6.1 C-CAST.....	12
2.6.2 FLAME2008.....	12
2.6.3 MyTV 2.0.....	15
2.6.4 Notification Service for DVB-H Mobile Broadcast.....	16
2.7 IMPLEMENTATION TECHNOLOGIES	17
2.7.1 HTTP REST.....	17
2.7.2 Java Servlet.....	18
2.7.3 Java ME.....	19
2.7.4 MySQL.....	19
3 DESIGNING CONTEXT-AWARE MOBILE SERVICES AND APPLICATIONS UTILIZING DVB-H	20
3.1 DEFINING CONTEXT-AWARE MOBILE TV SERVICES AND APPLICATIONS	20
3.2 USER ACCEPTANCE OF MOBILE SERVICES AND APPLICATIONS IN DVB-H.....	21
3.3 DESIGN CHALLENGES OF CONTEXT-AWARE MOBILE SERVICES AND APPLICATIONS	22
3.4 HOW TO DESIGN CONTEXT-AWARE MOBILE SERVICES AND APPLICATIONS	23
3.5 CONTENT RECOMMENDATION IN CONTEXT-AWARE DVB-H MOBILE SERVICE	26
4 DEVELOPED SOFTWARE FRAMEWORK	28
4.1 FRAMEWORK REQUIREMENTS	28
4.2 GENERAL VIEW OF CONTEXT-AWARE DVB-H SERVICE SOFTWARE FRAMEWORK.....	31
4.3 METADATA DESCRIPTIONS	34

4.4	OBTAINING USER INFORMATION, PROFILE AND CONTEXT	35
4.5	ACQUISITION OF CONTENT METADATA AND CONTENT SCHEDULING	36
4.6	THE RECOMMENDATION INFERENCE PROCESS.....	36
4.6.1	<i>Inference Engine</i>	38
4.6.2	<i>Recommendation Filter</i>	39
4.6.3	<i>Recommendation Pool Manager</i>	40
4.7	CLIENT APPLICATION	40
4.8	SUMMARY OF THE DEVELOPED SOFTWARE FRAMEWORK.....	41
5	REFERENCE IMPLEMENTATION.....	43
5.1	IMPLEMENTATION-SPECIFIC BACKGROUND TECHNOLOGIES.....	43
5.1.1	<i>Delivery Context Service</i>	44
5.1.2	<i>DVB-H scheduling using Sofia Digital Backstage Service Manager</i>	46
5.2	INTEGRATED REFERENCE IMPLEMENTATION	46
5.2.1	<i>Integration management interfaces</i>	51
5.3	REFERENCE IMPLEMENTATION SCENARIOS	52
5.3.1	<i>Basic demo scenario</i>	53
5.3.2	<i>Museum demo scenario</i>	56
5.3.3	<i>Car demo scenario</i>	57
5.3.4	<i>Predownload demo scenario</i>	60
5.4	VERIFICATION OF SOFTWARE FRAMEWORK REQUIREMENTS	62
5.5	SUMMARY OF THE IMPLEMENTATION	63
6	DISCUSSION OF THE DEVELOPED SOFTWARE FRAMEWORK AND REFERENCE IMPLEMENTATION.....	65
6.1	EVALUATION	65
6.2	CONSIDERATIONS AND FUTURE IMPROVEMENTS	68
6.3	SUMMARY OF THE DISCUSSION	72
7	CONCLUSIONS	73
	BIBLIOGRAPHY.....	75
	APPENDICES	79

ABBREVIATIONS

API	Application Programming Interface
ATSC-M/H	Advanced Television Systems Committee – Mobile/Handheld
CBF	Content-Based Filtering
CF	Collaborative Filtering
CGI	Common Gateway Interface
CLDC	Connected Limited Device Configuration
DC	DVB-H Client
DRS	DVB-H Recommendation Service
DVB-H	Digital Video Broadcasting - Handheld
DVB-T	Digital Video Broadcasting – Terrestrial
ESG	Electronic Service Guide
ETSI	European Telecommunications Standards Institute
FEC	Forward Error Correction
FLUTE	File Delivery over Unidirectional Transport
GPS	Global Positioning System
GUI	Graphical User Interface
HTTP	Hyper-Text Transport Protocol
IETF	Internet Engineering Task Force
IMEI	International Mobile Equipment Identity
IP	Internet Protocol
IPDC	Internet Protocol Datacast
LAN	Local Area Network
MIDP	Mobile Information Device Profile
MIME	Multipurpose Internet Mail Extensions
MPE	Multi-Protocol Encapsulation
MPEG	Moving Picture Experts Group
RDBMS	Relational Database Management System
REST	Representational State Transfer
RPM	Recommendation Pool Manager
RTP	Real-Time Transport Protocol
SDBSM	Sofia Digital Backstage Service Manager
SMS	Short Message Service
SQL	Structured Query Language
T-DMB	Terrestrial Digital Multimedia Broadcast
TPS	Transmitter Parameter Signaling
TS	Transport Stream
UDP	User Datagram Protocol
UI	User Interface
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

1 INTRODUCTION

1.1 Background and motivation

Since the dawn of mobile communications, when first generation (1G) networks were implemented in 1984 [1], mobile users – and along with it mobile services – have been increasing exponentially. Now, at the brink of implementing fourth generation (4G) networks enhancing data rates to around hundreds of megabits, mobile services, using the most sophisticated technology available, need to be studied in order to understand how they meet the users' needs.

During the recent years it has been more a rule than an exception to mention the words *context* and *context-awareness* when discussing the most prominent issues regarding mobile device user experience. The evident overflow of information that is taking place in current digital network environments has necessitated individual consumers, or users, to learn, adapt and filter incredibly diverse and massive amounts of information on wide range of varying user interfaces.

Thus, to ease the strain falling upon the users, the concept of context is brought up when designing mobile services that need to understand the needs of individual users and to offer the highest possible quality of service, i.e., right content in right situations. These challenges have been more and more at the core of mobile service research since the middle of the 1990s, when Schilit and Theimer [2], for the first time, defined the term context. Additionally, the mobile devices themselves started to become commonplace during the 1990s. More recently on several years, Gartner's Emerging Technologies report [3] has recognized context-awareness as one of the most substantial technologies in the future.

By using the knowledge regarding user's contexts it is possible to design context-aware services and applications, whose major objective is to enable the end-user devices to act intelligently in a multitude of situations and environments. To date, a lot of research has been conducted to enable context-awareness in mobile services, but so far only few actual applications have been deployed, as Zhang *et al.* [4] conclude. The biggest reason for this is explained with the gap between the technical understanding and the human perception of context information. Studies related to context-aware applications are also essential, as Hiltunen *et al.* [5] mention. To design successful context-aware applications, users' perceptions towards the topic are needed. The relevant issues include how a user distinguishes contextual information sources and their categorization, and the meaning and needs people propose for context-aware applications.

As the competition for the users' attention in the mobile sphere markets tighten, cost-effective content delivery methods become essential. This is where *mobile TV* comes in. The most prominent technology behind mobile TV is *Digital Video Broadcasting – Handheld* (DVB-H). The subscriptions related to this broadcast technology have been estimated to increase globally to 171 million in the year 2010 [6]. The revenues from

those subscriptions are expected to be around 10 billion €[7]. DVB-H mobile TV services have already been launched in over 10 countries and in about 30 more there are currently completed or on-going trials [8]. DVB-H is the recommended standard for mobile TV in Europe as endorsed by the European Commission in 2008 [9].

The fact DVB-H is a broadcast medium makes it able to provide services simultaneously for a large number of people, which consequently makes it a very cost-efficient technology. Furthermore, this kind of digital broadcasting utilizes high bandwidth channels and high transmission speeds, which results in high quality and wide selection of content. Compared to point-to-point connections, broadcasting is able to deliver media content virtually to a limitless amount of users simultaneously. DVB-H utilizes Internet Protocol Datacast, which grants access to the distribution of any kind of digital content, including applications. Furthermore, it provides interactivity to mobile TV using DVB-H. This enhances the types of content that can be broadcast for mobile reception. It is apparent that mobile TV has a lot of potential to become the biggest player in the content delivery industry. [10]

To triumph, DVB-H must be able to adapt to the trends of the mobile sphere. Taking into account the fact that currently mobile service users demand more and more *personalized* content, and that the service providers incapable of delivering this will most likely be unsuccessful, DVB-H technology necessarily needs to incorporate its share of the context-awareness business. However, a major problem with this technology is that it is inherently not meant for individually personalized content as it is a broadcast technology by nature. At best one could imagine the content being broadcast is only what the *majority* of the people – instead of each individual – want to see.

Mixing all of the above requirements related to context-aware mobile services and the massive financial potential of DVB-H, it all boils down into a question of *how to make DVB-H more context-aware*. When users are mobile, their context of use changes much more rapidly than when they are stationary – location changes, the devices and the people they interact with change more frequently, and users' goals and needs change. This inflicts the mobile computing platform with the need to sense potentially rapidly changing context, synthesize it and act upon it. [11] While completely personalized content for DVB-H subscribers may sound science fiction at the moment, it may very well become reality in the future. Meanwhile, it is important to strive towards this goal step by step. The purpose of this Thesis is to take that step forward. The designing process of Context-Aware DVB-H Service, described in this Thesis, began when in CoDe-project [12] it was deemed necessary to design and implement a context-aware mobile service, which would utilize DVB-H as the content delivery medium.

1.2 Research questions and goals

This Thesis describes the challenges that arise from combining DVB-H technology with personalized content and context-aware mobile services and applications. Consequently, the most essential question faced is: *in what way can context-awareness and personalization be utilized in recommending DVB-H-mediated content to users*. To answer this

question, DVB-H technology and various concepts related to context-awareness and personalization, and the challenges related to designing context-aware mobile services and applications, along with some solution proposals, are presented. Guidelines from the solutions along with novel ideas from some earlier work are utilized in designing a DVB-H service software framework, which exploits context-awareness and personalization to offer content for its users. This framework is then realized by constructing a functional server-client reference implementation, which is able to make recommendations for the user about any DVB-H-mediated content. This content is most likely something the user is actually interested in.

1.3 Organization of the Thesis

The next Chapter presents in detail all that is necessary for building a context-aware DVB-H service software framework: DVB-H technology, context and its representations, context-awareness and personalization, and some earlier work related to context-aware mobile services. At the end of the 2nd Chapter, relevant technologies related to the reference implementation are also presented. The 3rd Chapter first familiarizes the reader with the concepts of context-aware mobile services and applications. After this, typical mobile TV usage scenarios are described based on earlier research. Now understanding the specific way of using mobile TV services and applications, the challenges that will be faced when designing a context-aware mobile service or application, especially when utilizing DVB-H as the delivery method, are examined in the remaining Sections. In the 4th Chapter, the requirements for designing a context-aware DVB-H service are elaborated. Substantially, the 4th Chapter describes in detail the developed software framework. The 5th Chapter demonstrates the actual reference implementation, accompanied with several demo scenarios, which validate the desired functionalities and the set requirements of the developed framework. The results are then discussed and evaluated in the 6th Chapter. Additionally, possible drawbacks and future improvements are also considered. 7th Chapter concludes the Thesis with an overview of the accomplishments.

2 BACKGROUND

To start investigating how DVB-H technology and context-awareness could be combined in practice, the definitions and concepts behind these two relevant factors are introduced in this Chapter. First, DVB-H is considered to the necessary extent of this Thesis. It is followed by defining the concept of context and how it can be represented using context representation models. When it is clear what is meant by context, the essential term, context-awareness, is elaborated. After this, personalization and adaption of content are explained. The former of these can be considered to be essential when designing a context-aware mobile service. Finally, after earlier work related to context-aware mobile (TV) services is introduced, relevant technologies related to the reference implementation are introduced. The mentioned earlier work is later on utilized in designing the software framework presented in this Thesis.

2.1 DVB-H technology

DVB-H is one of the technology standards behind mobile TV. It is used mainly in Europe and Asia. Other standards also exist, such as Advanced Television Systems Committee – Mobile/Handheld (ATSC-M/H) in North America and Terrestrial Digital Multimedia Broadcast (T-DMB) in South Korea. DVB-H can be described as follows:

DVB-H is a technical specification for the transmission of digital TV to handheld receivers such as mobile telephones and PDAs. [9]

DVB-H technology is derived from the widely adopted standard for digital terrestrial TV, also known as DVB-T. DVB-H is designed to operate with a limited battery life and in difficult reception conditions, such as while moving in a vehicle at high speed. The usual applications for DVB-H are mobile TV, video streaming in general and file downloads. [9]

DVB-H delivers IP-encapsulated data in terrestrial networks by using the DVB Internet Protocol DataCasting (IPDC) specification, which enables the broadcasting of any kind of digital content. Additionally, IPDC enables the receiver to communicate with the sender via some different communication channel, which adds interactivity to the DVB-H usage paradigm. [13] However, these interaction schemes are typically proprietary and in most cases limited to ordinary SMS text messages [14]. The data in DVB-H network is delivered by using two different mechanisms: for streaming of audio, video and subtitles, Real-Time Transport Protocol (RTP) is used over User Datagram Protocol (UDP) and Internet Protocol (IP), while for file download services, File Delivery over Unidirectional Transport (FLUTE) protocol is used [14]. These are the two essential delivery modes in DVB-H.

In a technical specification [15] by European Telecommunications Standards Institute (ETSI), the Electronic Service Guide (ESG) in DVB-H is described. ESG contains information about the services available. By using the information in ESG, a user can select the services and items she is interested in, and find stored items on the terminal.

ESG operations take place after DVB-H receiver has been initialized and the terminal is synchronized to a particular transport stream carrying IPDC services. Based on the ESG information rendered to a user through an ESG application, a specific service can be selected. The ESG also provides information, which enables the terminal to connect to the related content IP stream in the DVB-H transport stream. The most important information in an ESG is Service, ScheduleEvent, Content and Acquisition fragments. Respectively, these describe essentially the following: IPDC service (i.e., channel) information; broadcast time of a scheduled item, which is a content item of a service; metadata for describing the content, and information to access a service or content. One part of the Acquisition fragment is Session Description Protocol (SDP), which contains information the terminal needs in order to be able to receive and consume the content of a service. To start the consumption of a DVB-H service, the terminal first needs to receive the ESG via a FLUTE session. Having received the ESG, user then browses the rendered ESG and chooses a service she wants to consume. SDP-file specific to the chosen service is then utilized by the client application to start receiving and consuming the wanted data from the service.

Since DVB-H in some respects is backwards compatible with DVB-T, the two technologies can share the same multiplex. One multiplex translates into a Transport Stream (TS). What is a transport stream? One DVB network is a collection of MPEG-2 transport streams. A single transport stream is composed of DVB services, which practically represent the “channels” of TV in the traditional meaning. Essentially and specifically, in DVB-H a DVB service can transport any type of data. This relation between DVB network, transport stream and DVB service is depicted in Figure 1.

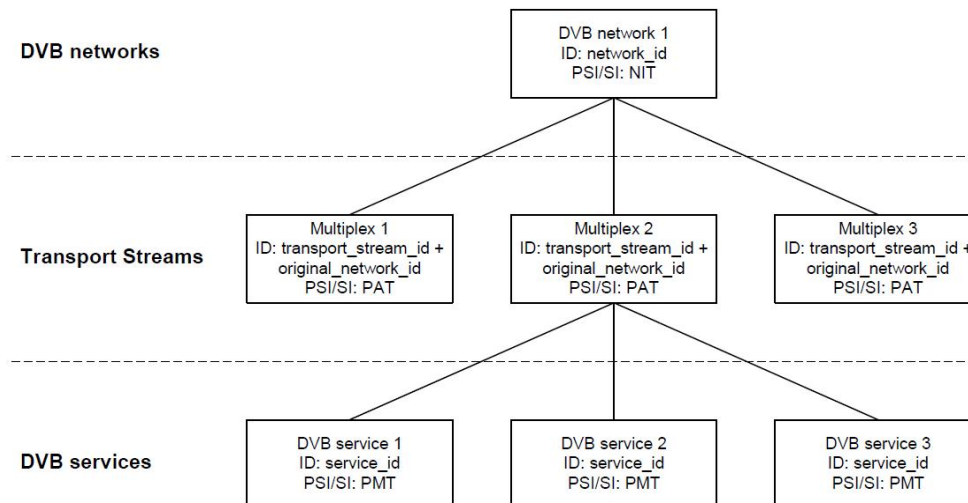


Figure 1. Relation between DVB networks, Transport Streams and DVB services. DVB network is composed of multiplexes, which in turn are composed of DVB services. [16]

In DVB-T a multiplex is composed of MPEG-2 TV services and Multi-Protocol Encapsulation (MPE) section streams. MPEG-2 TV services correspond to the channels mentioned before. MPE section streams are basically any IP packet data in a transport stream. Even though MPEG-2 TV services in DVB-H can be transported in the same

multiplex with DVB-H data thus making DVB-T compatible with DVB-H, all DVB-H data is sent using MPE sections. This means everything in DVB-H is IP-based. The way, in which the IP datagram packets are contained into MPE sections and furthermore into TS packets, is depicted in Figure 2.

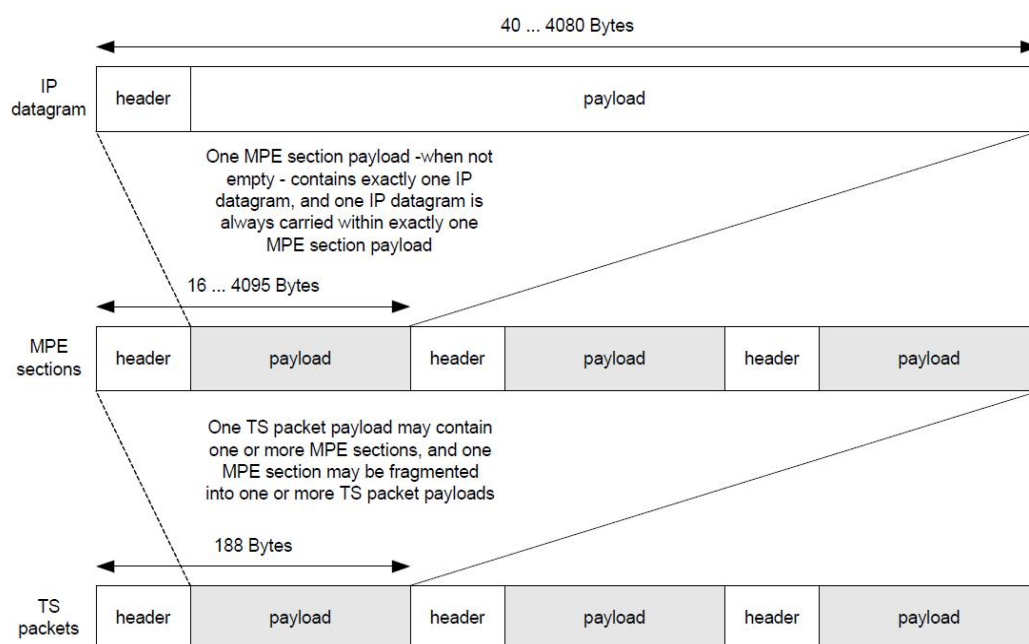


Figure 2. Relation between TS packet stream, MPE section stream and IP stream in DVB-technology. Each of the streams has a header part, but also a payload part, which carries the higher-level stream. Transport Stream packets contain MPE sections and MPE sections carry IP datagrams. [16]

Figure 3 represents the general view of how DVB-H differs from DVB-T and how the transport streams are composed and broadcast. The enhancements in DVB-H compared to DVB-T – also visible in Figure 3 – are explained next. To improve the robustness and the mobility of the signal, DVB-H incorporates an additional Forward Error Correction (FEC) scheme to the MPE sections. The use of FEC results in improvements related to Carrier-to-Noise and Doppler performance, and to better tolerance against impulse interference. These phenomena are mainly related to the mobility of a mobile device. Furthermore, DVB-H includes time-slicing functionality, the objective of which is to reduce the average power consumption of the terminal and to enable smooth and seamless service handover. Here individual DVB-H services are sent as high bit rate data bursts in small time slots. Time-slicing enables the receiver to stay active only for a very short time, i.e., while receiving these bursts. The data is buffered into the receiver device and can be consumed while the receiver is inactive between the bursts. Between the bursts receiver may monitor the neighboring DVB-H cells for making optimal service handover decisions. Time-slicing is essential for the receiver's battery life and thermal balance. Moreover, to improve network planning flexibility by trading off mobile reception performance and network cell size, an additional 4K mode has been introduced in DVB-H, which is a compromise between the traditional 2K and 8K modes. 8K mode allows large network cells but the DVB-H receiver can not move nearly as

fast as in 2K mode with smaller network cell size. The modes refer to the amount of active carrier waves in the signal. Finally, Transmitter Parameter Signaling (TPS) is used in DVB-H. It provides an easy access to signaling parameters for DVB-H receivers, thus speeding up service discovery. [9], [17]

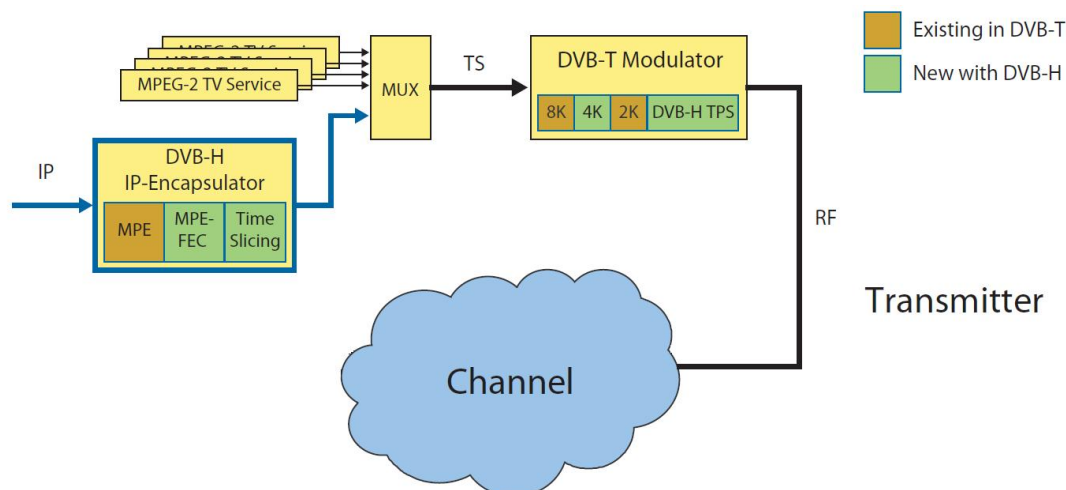


Figure 3. Additions to the DVB-T system by DVB-H. Multi-Protocol Encapsulation sections with added Forward Error Correction functionality is used for improving error resiliency in DVB-H. Time-slicing essentially minimizes the battery usage of the client device. Additionally, 4K mode offers a compromise between network cell size and the speed with which the terminal can travel. The DVB-H IP stream can be multiplexed alongside with the MPEG-2 TV Services of DVB-T. [9]

2.2 Context and its types

The use of a mobile device gives rise to dynamically changing situations, where the context of a user changes constantly. It is proper to say context is among the most relevant terms in the interaction between a human and a computer, because essentially its definitions are used to make computers aware and responsive to the individual needs of a user at any given time or place. In order to take advantage of this extended ability to communicate and interact, context needs to be defined. The most common definition for context was elaborated by Dey and Abowd [18]:

Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves (and by extension, the environment the user and applications are embedded in [19]).

Furthermore, in [18] Dey and Abowd state that the most important contexts are *location, identity, activity* and *time*, which constitute *primary context* types. *Secondary contexts* are then those contexts, which can be inferred by using the information from the primary context types.

Zhang *et al.* [4] categorize contexts in a mobile environment into five groups: 1) user context, such as personal time schedule or user preferences; 2) device context, such as screen size or operating system; 3) activity context, such as task type or time to perform an activity; 4) network context, such as network bandwidth, and 5) external physical environment context, such as temperature or noise level in the surroundings. Additionally, in some papers (e.g., [20] by Gustafsson) contexts are divided into low- and high-level contexts, which seem to correspond respectively to the classification of primary and secondary contexts by Dey and Abowd in [18]. Low-level context refers to unprocessed context information received from a context source. This information can be, for example, physical, technical or social. Example of low-level context information includes indoor and outdoor location coordinates by using 1) Global Positioning System (GPS), 2) network cellular ID or 3) Bluetooth / Wireless LAN hotspots. Additionally, time, identity of the user, or the presence of other people can be used as low-level context information. High-level context is information inferred by the system by using the low-level context information. It is an assumption of “what is going on” at the moment. Knowledge about user’s goals helps prioritize the device actions and select the most relevant information sources.

One more way of categorizing contexts is by dividing them into external and internal contexts. External context is a physical context that can be sensed with the sensors of the device. Internal context is either defined by the user herself or inferred automatically by following the actions of the user. [21] Evidently, there is a wide array of context categorizations available for the designers of context-utilizing systems.

2.3 Context representation models in mobile systems

For representing context information in unambiguous, uniform and machine-readable way, we need to define a method for describing the concepts that are relevant for a particular mobile application. Currently, there are several ways of accomplishing this. The use of *ontologies* is the most prominent method for representing “information about information”, i.e., metadata. In addition, other information representation models do exist, such as the *markup scheme model*.

As explained by [22], “ontology provides a shared vocabulary, which can be used to model a domain”. Another, perhaps more prominent definition for ontology is by Borst [23]: “An explicit formal specification of a shared conceptualization”. According to [24], a formal ontology consists of 1) facts representing explicit knowledge, which comprises concepts, their properties, and instances representing the entities described by concepts, and 2) axioms and predicates representing implicit knowledge, to which semantics are added and from which knowledge from facts on demand is derived.

Another method for context representation is markup scheme model. It uses a hierarchical data structure that consists of markup tags with attributes and content to model context. Typical way of utilizing markup scheme model is to describe metadata by using eXtensible Markup Language (*XML*). XML is a textual language used to describe

arbitrary data structures in machine-readable form by utilizing documents called *XML Schemas*. By using XML Schemas it is possible to pre-determine the format, vocabulary and the structure of an XML document. Consequently, this forces the designers of an application, which utilizes the XML Schema, to comply with the requirements of the given XML Schema. Hence, all the documents complying with the XML Schema are unambiguous to both machine- and human readers. XML Schemas are well-suited in the mobile service landscapes due to their computational lightness.

In [4] Zhang *et al.* compare various context representation models. Representing context metadata in an *ontology-based model* enables contextual knowledge sharing and context reuse in a ubiquitous computing system. Unfortunately, in many cases ontologies are unable to enumerate all possible context events that can trigger actions, and in most cases ontologies are not publicly available. Based on the experiences of the author, in practice and in many cases, the problem of using ontology-based context representation model in mobile services is its complexity. Ontologies are able to describe very intricate information; the problem within is the complexity of the information description that soon can become too heavy for the mobile devices. Zhang *et al.* continue by evaluating markup scheme model. The model's advantage is the ability to cover higher dynamics of contextual information, but then again it may be difficult and non-intuitive to capture complex contextual relationships and constraints, which can be more easily described in ontology-based models.

2.4 Context-awareness: inferring situation from sensor data

From a practical perspective, the concept of *context-awareness* is at least equally as important as context. Context-awareness enables computers to understand the reason for a specific situation of a user. Dey and Abowd [18] define context-awareness as follows:

A system is context-aware if it uses context to provide relevant information and/or services to user, where relevancy depends on the user's task.

What are the steps in the actual process of inferring user context? The process, which begins from plain context-sensing all the way to proper understanding of the actual situation of the user, is described in [4] by Zhang *et al.* To start the process of understanding the context of a user, context information needs to be collected. In [4], sensing sources are categorized into physical sensors, virtual sensors and logical sensors. Physical sensors, or hardware sensors, detect information from the physical world. Virtual sensors, or software sensors, detect information about the networked, or virtual, world. Logical sensors provide a higher level of information abstraction, such as logical deduction or, i.e., distance from an event location. Logical sensing more or less corresponds to the secondary context in [18] and internal context in [21]. These three methods of sensing context are utilized in the first phase of inferring the situation of a user. The phase is called *context acquisition*.

The next step is *context abstraction*, the idea of which is to make the context sensible for an application, e.g., transforming the context information into a format the applica-

tion can understand. This can be done, e.g., by utilizing context representation models, which define the representation format of the information. The third step in the process is *context interpretation*: “the process of transforming one or more context sources into a new, understandable piece of context information”. Context information is analyzed to make a prediction for obtaining semantics behind correlative context features. For a context-aware system this means the context must be in a structured, uniform and interchangeable format to allow complex rules to be written and easy integration of context. Unfortunately to this day, there exists no standard model for context representation. Due to the complex requirements of context models for context-aware systems, the previously described ontology-based model is the most promising model for context representation [25].

The following step in [4] is *context aggregation*. By fusing data from different sensors it is possible to select and integrate the only context information relevant to an application. This means irrelevant context information is forfeited. Finally, *context inference* aims to augment mobile devices’ cognition capabilities and interaction with users. Context inference can be considered as system adaptation based on real-time sensing of a user’s context. It can be conducted either through manual user input and setting, or implicitly, where inferred awareness is based on the interaction occurring between the user and the system.

2.5 Personalization and adaptation of content

An efficient model for handling, sharing and storing context information is essential for a context-aware system. The semantics of context information is used as a basis for *adaption* and *personalization*, which help the user to get relevant content in a relevant format for the current situation. [26] In [27] Laakko & Gustafsson describe personalization. Considering a single user, the objective of personalization in practice is to match the offered content and services to the user’s personal preferences as well as possible. The purpose is to improve the user’s experience of a service by offering relevant content suited for the particular situation. Personalization involves a process of gathering user information during interaction with the user.

According to Dey and Häkkinen [11], user profiles are entities containing information regarding the user and her preferences, which can be used to offer useful information for profiling or personalizing services or refining information retrieval. Zhang *et al.* [4] elaborate this by mentioning that the idea is to collect the information either directly through user solicitation or indirectly through explicit or implicit user feedback. This user context is then stored in a file called user profile, which is used to adapt the system’s behavior. Weissenberg *et al.* [24] mention the availability of *user profiles* as the basis for personalization. Personalization can also be based on semantically refined user context information. As an example of personalization, the personalization of advertisements can be used to increase the effectiveness of advertising by reaching the users who most likely would be interested in them [28]. Similarly, TV content can be recommended to viewers, whose interests or needs are uniform to the content offered.

According to a paper by Choriantopoulos [28] the two most popular approaches to personalization are: 1) Collaborative Filtering (CF) and 2) content-based filtering (CBF):

Collaborative filtering is based on the assumption that users who have agreed in the past in their subjective evaluation on observed items will eventually agree in the future.

Content-based filtering is an information retrieval technique that makes predictions upon the assumption that a user's previous preferences are reliable indicators for future behavior.

Collaborative filtering involves a social aspect whereas content-based filtering is more personal to a specific user. In a paper by Fong *et al.* [29] the differences between content-based and collaborative filtering are explained in more detail. The idea behind CBF is to select items based on the correlation between the content of the items and user preferences. Typical problem of CBF is inherent: it can only recommend items with high scores with regard to the user profile. This means the user is restricted to seeing only items similar to those already rated; no new items will be ever recommended. On the other hand, CF is based on the similarity between currently active user and other users. Its objective is to find new items the active user has never seen but which are inferred as “interesting” to her. This inference is carried out by using the ratings and recommendations of the other users with similar interests in regard to those of the active user. CF has two approaches: item-based CF and user-based CF. In the former, if content A and content B are liked by the other users, then if the active user likes content A, it can be inferred that content B would also be interesting to the active user. In the latter, if content A is liked by the active user and also by the other users, and if the other users like content B as well, we can infer that the active user would like also content B, as she has similar taste in preference with the other users.

Zhang *et al.* [4] elaborate the concept of content adaptation by first describing the challenges related to various limitations of mobile devices. It is mentioned that Web information is poorly suited for mobile devices: the unreliability of wireless networks, user mobility and physical constraints of mobile devices impede with mobile users' attempts to take advantage of the information on the Internet. Relevant question therefore is how to adapt Web content to meet the needs of users, fit the characteristics of individual devices, and adjust to dynamic external context. For adapting content to a mobile device, user and device profiles along with physical context may be used to support this kind of adaptation. The physical and virtual characteristics of the device, such as screen size, memory size and supported MIME-types, as well as the quality of service the user expects are examples of relevant factors in adapting content. Therefore, content adaptation is used to improve the users' experience of digital content.

2.6 Earlier work

A significant amount of research has been conducted previously in the field of context-awareness, especially related to mobile services. When restricting the field of research to combine both mobile context-awareness and DVB-H, it seems there is still a lot of research to be carried out, since this is a rare research topic. In the following, four explorations into the field of mobile context-awareness and DVB-H are presented. C-CAST shortly delves into group contexts in one-to-many mobile services. FLAME2008 offers a perspective on a framework, whose operating principles might be of use in designing a context-aware DVB-H service. MyTV 2.0 and Notification Service for DVB-H Mobile Broadcast serve to show how DVB-H can be made context-aware by employing ESG and notifications, respectively.

2.6.1 C-CAST

In the project called C-CAST [30], or Context Casting, the objective was to design a context distribution framework which would support the collection, processing and management of context information. From this information, group context and situation context could then be inferred. The main delivery mechanism was multicasting, which enables the efficient use of network in one-to-many services. The traditional concept of subscription based groups was in C-CAST extended with situation-oriented groups.

The key hypothesis in C-CAST was that by combining context-awareness and mobile multicasting technology it would be possible to provide an end-to-end context-aware communication framework, which would make context-aware mobile services commercially attractive. Both applications and the network entities utilize the context management framework. Substantially, the target group consumers in C-CAST are provided with relevant information by grouping the users in a specific location according to their context and preferences. These groups are called *multicast groups*, to each of which suitable content can be delivered. For example, in a mall the user's gender or age affects into which group the user is assigned to. She would then receive relevant advertisements based on her context. C-CAST improves the way multicast broadcast content is delivered in a context-aware way to the user, resulting in efficiency of the network usage. Relevant solutions in C-CAST in respect to context-awareness are the use of context-aware groups, which enable the efficient use of the network in one-to-many services, such as DVB-H, and still offer moderately relevant content for individual users.

2.6.2 FLAME2008

For understanding the idea behind FLAME2008, push and pull technology need to be defined:

Push technology, or server push, describes a style of Internet-based communication where the request for a given transaction is initiated by the publisher or central server. [31]

Pull technology or client pull is a style of network communication where the initial request for data originates from the client, and then is responded to by the server. [32]

To customize services, individual situation information and personal demands need to be accounted for, as there is a vast amount of Web services and information available today. For enabling flexible service selection on a semantic level, semantic descriptions of situations and offered services are needed. This is what Weissenberg *et al.* [24] strived for in FLAME2008. The idea in FLAME2008 is an individual push of meaningful offers (with pull possibilities) for information and services to the mobile frontend, based on the current context, situation and profile of each user. Also, a personalized semantic search for information and services is provided. Situation is derived using various sensors and user profile information. *Offers* and *demands* are matched against each other using semantic technologies.

Considering the offer, service providers may bind their services into so-called situation profiles. These situations and services consist of a set of structured attributes characterizing the situation or service. The attributes use semantic categorizing, which means the attributes are taken from a set of semantic concepts in an ontology. Demand, on the other hand, refers to a user-maintained user profile (set of properties) and low-level sensor data, which are both accessed by the FLAME2008 inference engine, either on-demand or as part of a query. From this information, a situation request profile is constructed. It is then matched against the profiles of all situations known by the system. Similarly, a service request profile is constructed and matched against all registered service profiles. Then, as significant changes are detected, the new relevant services and information are pushed to the user's mobile device.

FLAME2008 architecture is depicted in Figure 4. It consists of an information-logistics engine, a content broker, a user profile and a context component and a service roaming component. Information-logistics engine is responsible for personalized push and pull mechanisms for information and service offers, which are based on semantic matching of services and situations. Content broker provides content access – services included – and maintains a semantic registry, which uses the agreed application ontology for describing information. User profile and context components provide user-specific information especially for situation detection. Finally, service roaming component is designed to overcome the problems of the services' possibly limited (e.g., regional) scope: as long as the user's situation is valid in terms of context, service may be roamed, regardless of the original geographical scope of the service.

The semantic situation and service matching and the way the service selection process proceeds are presented in Figure 5. Situation detection is illustrated on the left-hand side. Context data from sensors is imported to the inference engine, while the gazetteer mechanisms derive instances of the higher-level ontology concepts, which results in an abstract context with logical higher-level context values. This abstract context with user profile information, and other sources, is used for constructing the situation request profile. This situation request profile is then matched against all the situation profiles known by the system. The described process leads in to a set of situations that fulfill the

request profile. A user may be in any of these situations. On the right-hand side of Figure 5, the implicitly constructed service request profile is matched against the profiles of all known services found in the semantic registry. This service request profile then refers to the matching situations of the situation-set acquired previously. This is how the users receive relevant services in FLAME2008.

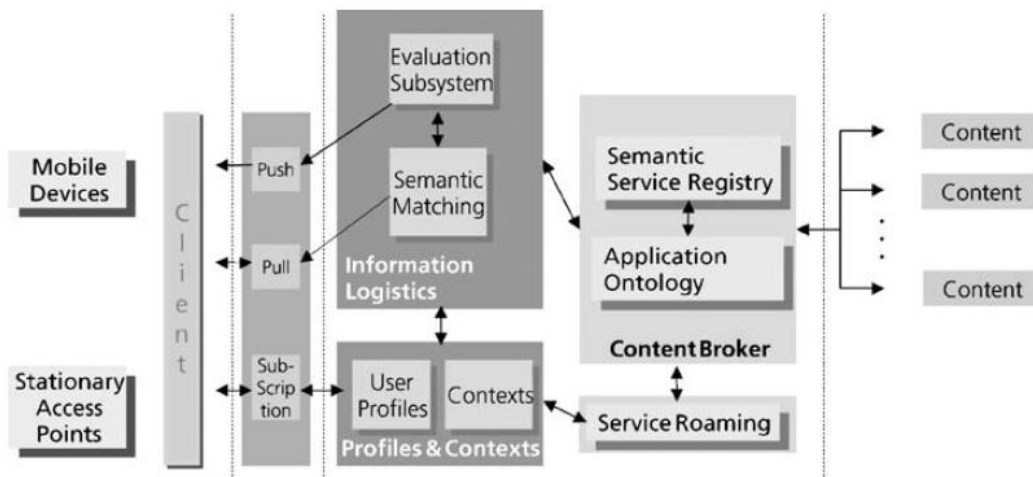


Figure 4. FLAME2008 general system architecture. Information-Logistics is responsible for offering relevant content to the subscribers of the service. The subscribers share their user profile and context with the service. Information is described and accessed via ContentBroker. [24]

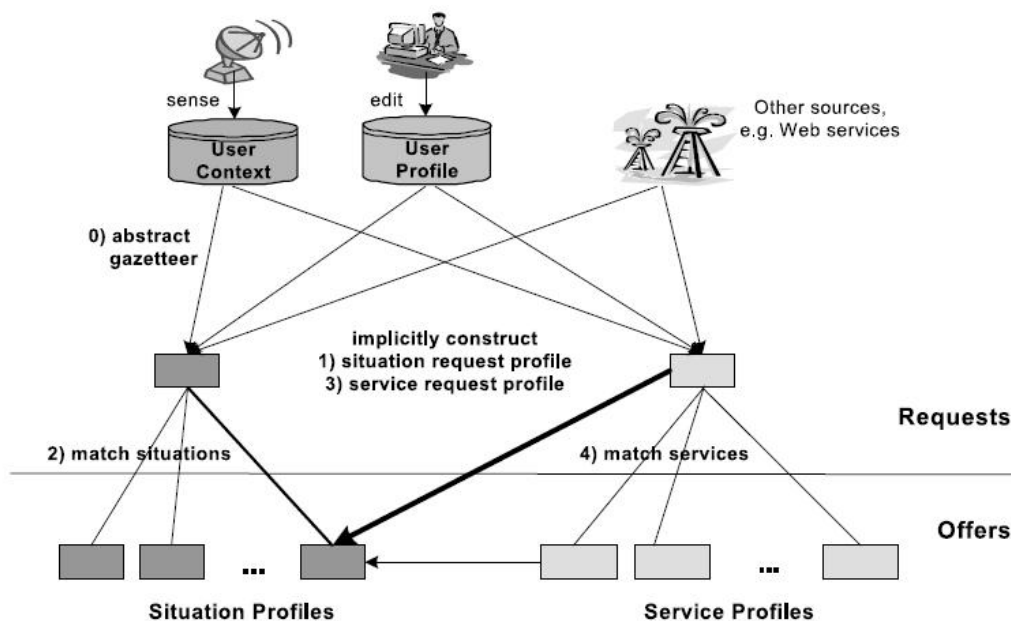


Figure 5. Semantic situation and service matching mechanism in FLAME2008. Services and situations relevant to the users are matched with corresponding profiles generated in the service. [24]

2.6.3 MyTV 2.0

Utilizing collaborative tagging and semantic recommendations, the aim in MyTV 2.0 [33] is to offer interesting content for the users via mobile TV. According to the authors, a filtering system for selecting interesting content is needed because of three reasons: 1) multitude of available TV services; 2) accessing the services is very flexible, i.e., it can be done anywhere and anytime, and 3) mobile TV is generally viewed very briefly, between other activities. For this, collaboration and knowledge sharing between the users is introduced: the subscribers of mobile TV already know what content is interesting. This leads into the filtering not only having a social interaction aspect, but it will become more efficient as well. A formal semantic reasoning logic for knowledge inference is also implemented to utilize the information the users generate collaboratively.

The general architecture in MyTV 2.0 is composed of content providers, the MyTV providers and the MyTV users (Figure 6). The function of content providers is to tag content appropriately and offer it to the MyTV providers. Both tags and content are broadcast to the DVB-H network. In addition to ESG-metadata, an ontological description, namely *TVOntology*, describes the content more accurately, enabling all parties involved to share a common understanding of the keywords.

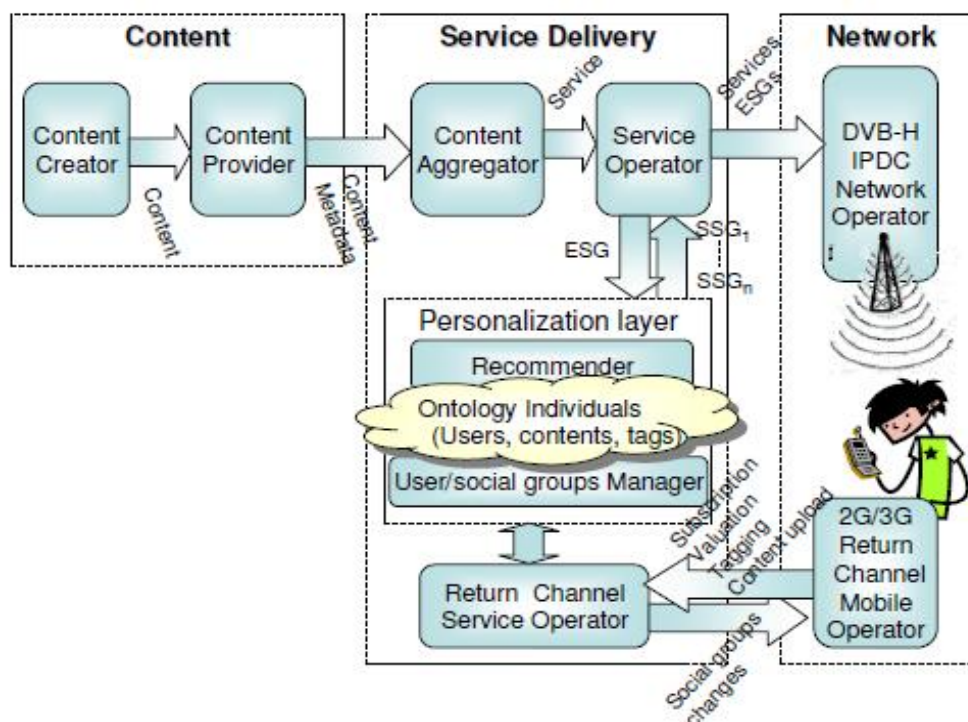


Figure 6. myTV 2.0 architecture composing of Content, Service Delivery and Network entities. The user utilizes 3G return channel for sharing and tagging relevant information needed for the service to evolve itself by extending ontologies. Also, content providers participate in this process by providing their own metadata. Specific ESGs are created for the DVB-H network to be able to make good recommendations for the users of the service. [33]

The MyTV users are in essential role, since they can record, tag and upload their own content to the MyTV provider via 3G-networks. Tags may be informal, but nevertheless they are collected in a folksonomy, which the users collaboratively and progressively build. In short, TVOntology provides the structure for the language used, whereas the emergent folksonomy conceptualizes a free vocabulary for additional subjective properties. Consequently, the MyTV provider processes two kinds of content: formally labeled contents from content providers and informally tagged content from subscribers. This information is used to form social groups. These groups are based on group filtering and have individual characteristics, according to which the content is broadcast to these groups. Finally, considering only one subscriber with specific characteristics, the received contents are re-filtered (individual filtering) in client device to display more relevant content for the subscriber. Individual filtering, however, is not required for the MyTV provider to be able to offer recommendations: the value in doing this is the freeing of resource-demanding tasks in the receiver device, as the inference is done completely server-side.

2.6.4 Notification Service for DVB-H Mobile Broadcast

Hornsby *et al.* [14] concentrated on designing a framework for a notification service, which addresses devices and users with messages, which may be related to emergencies, warning or other timely messages. This notification service is designed to be an extension to the DVB-H IPDC data transport system. The potential applications or notifications for this service include *service related information* such as change of schedule or availability, *location-specific services* such as regional news or weather forecasts, *specific services* such as news and sports reports, and *download, update or upgrade announcements* for, e.g., downloading software. Additionally, recently there has been increasing interest towards notification services especially in mass emergency cases such as earthquakes and tsunamis.

The notifications utilized in the framework “provide the means by which the network delivers messages about forthcoming events of interest to a terminal or user”. Once the notification is received by the end-user, the message may be presented either immediately or consumed later. User interaction is also possible, if the notification calls for actions from the user. This results in an evolution of DVB-H-based service consumption from passive to active consumption. The user is able to subscribe to specific service-related notifications, which enhances the relevance of the notifications for the user. The ESG used by DVB-H is extended with components, which carry information related to the notifications and are delivered using the FLUTE protocol in an IPDC-compliant manner. The notification architecture designed in [14] is a good example for future mobile multimedia broadcast services to enable more timely and interactive services and applications.

2.7 Implementation technologies

This Section presents the most prominent technologies, which could be utilized in realizing the practical implementation of the framework developed in this Thesis. In addition to using DVB-H as the media delivery channel, several key technologies might be feasible to apply in various parts of the practical implementation. Here, few potential technologies are shortly described:

- *HTTP REST* as an interface for decentralized server-to-server (inter-Servlet) communication and server-client communication.
- *Java Servlets* as the primary way of implementing reference implementation functionalities in the server-side.
- *Java ME* as the technology behind the application implemented in the client-side.
- *MySQL* as the reference implementation database for storing persistent information regarding the various used and content data.

2.7.1 HTTP REST

In the year 2000, Roy Fielding [34] introduced an architecture called Representational State Transfer (REST). It is a software architecture originally designed to utilize Hypertext Transfer Protocol (HTTP) for server-client communication. REST is based on widely used W3C- (World Wide Web Consortium) and IETF-standards (Internet Engineering Task Force) with support from multiple programming languages, Web servers and firewalls. Implementing REST is straightforward as the development is similar to dynamic Web pages and testing is possible with an ordinary www-browser. [35]

REST is based on few basic principles [35], [36]:

- The network services offered by a Web service, i.e., resources, are identified by an individual URI (Uniform Resource Identifier). A resource is the intended conceptual target of a hypertext reference [34].
- Resources are conceptually separate from the representations returned to the client.
- Resources have a common interface for reading, updating and deleting operations. PUT-command creates a new resource, which can be deleted with DELETE-command. The current state of the resource can be read with GET-command, whereas POST-command delivers new state information for the resource. In short, resources can be manipulated through these representations.

- State information is mediated via hypertext links. Communication with a resource is stateless, which means all the requests for a resource are independent. The response messages from a server contain URI-references to all possible and related resources.
- Messages are self-descriptive, which means each of them contain enough information to describe how to process the message.

2.7.2 Java Servlet

Java Servlets are platform-independent Java classes that run in a server application to answer client requests. Most frequently, HTTP is used as the client-server protocol for Servlets. Typically, Servlets are used for processing and storing data submitted by an HTML form, for providing dynamic content (e.g., database query returned to a client) and for managing state information on top of the stateless HTTP. [37], [38]

Some of the advantages using Servlets instead of the more traditional CGI (Common Gateway Interface) in adding functionality to a Web server include [38]:

- A Servlet does not run in a separate process. This removes the overhead of creating a new process for each request
- A Servlet stays in memory between requests. A CGI program needs to be loaded and started for each CGI request
- There is only a single instance which answers all requests concurrently. This saves memory and allows a Servlet to easily manage persistent data.

Servlets are managed by a *container*. Containers are Web server (e.g., Apache Tomcat by Apache Software Foundation) extensions that actualize the Servlet functionalities when interacting with Web clients via a request/response paradigm. These functionalities include network services, decoding of MIME-based requests and formatting MIME-based responses. A Servlet container is also responsible for managing the lifecycle of Servlets. [37]

A good example of a typical sequence of events related to Java Servlets is described in [37]:

1. A client (e.g., a Web browser) accesses a Web server and makes an HTTP request.
2. The request is received by the Web server and handed off to the Servlet container. The Servlet container can be running in the same process as the host Web server, in a different process on the same host, or on a different host from the Web server for which it processes requests.

3. The Servlet container determines which Servlet to invoke based on the configuration of its Servlets, and calls it with objects representing the request and response.
4. The Servlet uses the request object to find out who the remote user is, what HTTP POST parameters may have been sent as part of this request, and other relevant data. The Servlet performs whatever logic it was programmed with, and generates data to send back to the client. It sends this data back to the client via the response object.
5. Once the Servlet has finished processing the request, the Servlet container ensures that the response is properly flushed, and returns control back to the host Web server.

2.7.3 Java ME

Java Platform, Micro Edition (Java ME) is a platform that fits the requirements for mobile devices with a collection of technologies and specifications. Originally, Java ME was created for dealing with the constraints associated with building applications for small devices, which typically have limited memory, display and power capacity. [39]

Java ME technology is composed of 1) a *configuration* for providing fundamental libraries and virtual machine capabilities for a broad range of devices, 2) a *profile* with APIs (Application Programming Interface) for supporting narrower range of devices and 3) an *optional package* for more technology-specific APIs. The configuration used in mobile devices is the CLDC (Connected Limited Device Configuration), which comprises subset of the Java-class libraries. CLDC is targeted for resource-constrained devices. Profiles are subsets of configurations and specify more strictly the mobile device's Java framework for which the application is being developed. The most common profile is MIDP (Mobile Information Device Profile) defining, e.g., a GUI and a data storage API. It is commonplace to combine CLDC with MIDP to provide a complete Java application environment for devices with restrictions, such as mobile phones. Applications written for MIDP are called MIDlets. MIDlets can be run on any device conforming to the specifications of Java ME technology. [39]

2.7.4 MySQL

One concise description for MySQL can be found from [40]: “MySQL is a Relational Database Management System (RDBMS) that runs as a server providing multi-user access to a number of databases”. Database can be described as a structured collection of data. Database management system is needed to add, access and process data in these databases. The term “relational” refers to the fact that data is stored in separate tables instead of one big storage, which results in more speed and flexibility in database operations. MySQL itself is open source, which means the software can be modified by anyone. [41] For enabling Java Servlets to communicate with MySQL database, a connector is needed. In this reference implementation, “MySQL provides connectivity for client applications developed in the Java programming language through a JDBC driver, which is called MySQL Connector/J [(Java Database Connectivity)]” [42].

3 DESIGNING CONTEXT-AWARE MOBILE SERVICES AND APPLICATIONS UTILIZING DVB-H

How should context-aware mobile services and applications be designed? The services need to be able to offer content that is both relevant and interesting to the user. On the other hand, the applications must be designed in a way that results in positive overall user experience. Placing these challenges into the context of DVB-H content delivery, there are several issues that need to be addressed in order to design a well-functioning context-aware software framework utilizing DVB-H.

In this Chapter, the challenges related to designing context-aware mobile services and applications utilizing DVB-H are considered. First, context-aware mobile services and applications are defined. Following this, user acceptance of mobile services and applications from the viewpoint of DVB-H mobile TV typical usage habits is then explored to understand the challenges specific to this research area. With this in the background the challenges related to designing context-aware mobile services and applications are addressed. Finally, some guidelines for designing successful context-aware mobile services and applications are presented, especially with DVB-H in mind.

3.1 Defining context-aware mobile TV services and applications

According to Zhang & Zheng [1], *mobile service* can be defined as “a service that is available through mobile radio access at anytime and anywhere possibly through heterogeneous mobile devices”. Typically, a user interacts with a mobile service through a mobile device. *Mobile application* refers here to software applications, which run on a mobile device. No further specifications are included in the definition. The term “context-aware” here refers to the fact that both the service and applications are designed in a way that enables them to be context-aware. According to [20], the first prominent context-aware applications utilized location information as the primary means of inferring context, which is true also today. Later on, other context sources have also been employed for the benefit of context-aware applications. In [18] Dey and Abowd list context-aware features that context-aware applications may support: 1) presentation of information and services to user, 2) automatic execution of a service and 3) tagging of context to information for later retrieval. To support these features, all the three context sensing methods mentioned in Section 2.4 – physical, virtual and logical sensors – need to be utilized.

In [43] by Loebbecke *et al.*, four distinctions between various mobile services are pointed out: 1) one-to-one mobile services vs. one-to-many mobile services, 2) application mobile services vs. information mobile services, 3) user-controlled mobile services vs. user-received mobile services and 4) software industry mobile services vs. media industry mobile services. Related to 1), one-to-one mobile services require a point-to-point connection whereas one-to-many mobile services utilize, e.g., broadcast. In 2), application mobile services allow users to accomplish tasks by sending requests and by processing and storing data, while information mobile services provide users with information or entertainment. User-controlled mobile services in 3) sets the user in control

of what services are delivered, and requires, at least to some extent, interactivity. In user-received mobile services the user assumes a receptive role with very little or no interactivity. Finally, in 4) software industry mobile services' application cores originate from software industry, whereas media industry mobile services' information and entertainment cores originate from creative media industries.

3.2 User acceptance of mobile services and applications in DVB-H

User acceptance of any mobile service was extensively examined by Kaasinen [44]. According to the results, the focus in designing mobile services should be concentrated on providing key values to the user instead of implementing collections of useful features. Both personal and user-generated content are important in a successful service. These services should appear in the mobile client as seamless entities instead of separate services. Furthermore, a clear overview of the service, fluent navigation and smooth user interaction are all requirements for a good user experience. Personalized and situationally relevant services and information are of the utmost importance, although the effort towards personalization should be kept at minimum. Wide support for diversity of devices and networks are needed to support the ever-growing array of hardware. User must also be able to trust the services: in this context, personal information is used more and more, which means a user should be able to assess the reliability of the service. Finally, the actual values of the services must be communicated to the user in order for her to be able to realize the potential of individual services in everyday life. Zhang & Zheng [1] note that the users of mobile services demand more and more value added services. On the other hand, consumers' willingness to adopt and pay for a complicated and value added, novel mobile service declines. This means there is a clear mismatch between the introduction of new mobile services and consumers' willingness to adopt, accept, and use these services.

How is mobile TV used from the viewpoint of a consumer? In order to design widely accepted and successful mobile TV services, it is important to understand the aspects that affect how and where these services are consumed. Equally essential is to know how User Interface (UI) in these cases should be designed to make it as usable as possible. In their paper, Oksman *et al.* [45] describe how mobile devices are used in mobile TV context. Generally, mobile devices are used in three separate spheres: home, work and public. The latter is regarded as the typical usage sphere. Killing time while waiting, or keeping the user entertained or up-to-date while commuting are examples of typical use scenarios. According to another paper by Oksman *et al.* [46], short watching sessions are typical to mobile TV, although some longer durations of viewing have been observed. The most preferred content in mobile TV is news. It suits well the typical mobile TV using habits of an average user. Grobel [47] predicts customized services addressing specific interests of individual users will become important. Audio and video file sharing is considered to be one of the most significant phenomena in mobile TV [45].

User acceptance of mobile TV was investigated by Kaasinen *et al.* in [48]. The users evaluating mobile TV concluded that it was seen as easy to use and the value of it was

seen in topical and entertaining content. Mobile TV was seen as an extension to ordinary TV, freeing them from the bounds of the primary TV. This continuous access to TV was appreciated. Adoption of add-on, or interactive, services was hindered due to the short usage situations of typical mobile TV use. To enhance the adoption of these services, content, appearance or functionality familiar from other media should be considered.

3.3 Design challenges of context-aware mobile services and applications

The challenges related to designing context-aware mobile applications and services are substantial. The user acceptance is of utmost importance in the design process. Dynamic nature of mobile devices in varying situations, attempts to establish context-aware infrastructures and the design-gap between technical and social aspects are the main motivation for understanding how to design successful context-aware mobile applications and services.

According to a research conducted by Dey and Häkkinen [11], the users of context-aware mobile applications preferred applications that had higher degrees of proactivity, i.e., the application performing actions for the user. However, this resulted in the sense of losing control. In addition to this issue of control, the lack of feedback the application provides was seen as a problem: how would the user know what the application actually is doing and why. Third issue recognized was privacy: how is the context data of the user used by service providers to, e.g., track the user's preferences or location. Final concern for the users was information overload. Too many applications competing for user's attention at the same time might become very annoying.

In [11], Dey and Häkkinen also list usability risks in context-aware mobile applications. Fundamental cause of potential usability risk lies in uncertainty in context recognition. This affects substantially the designing of the user interface as it affects the selected features, their functionality and accuracy. Other usability risks are plentiful. Application complexity may affect the sensibility of the application if a proper level of feedback and transparency is not maintained. Lack of standardization in the field may lead to poor interoperability of services and applications. Related to this is the lack of commonly agreed ontologies or context representation models, which would alleviate the problem of subjective understanding of context attributes. The consequences from all these risks may lead to, e.g., spam notifications, unintuitive user interfaces, increased number of interruptions, poor categorization and attribute hierarchies, and unexpected device behavior.

Furthermore, in [11] Dey and Häkkinen identify four basic problems in building context-aware applications. The first problem is that context rarely comes from traditional devices, such as a keyboard or a mouse, i.e., from the devices the developers would have experience on. The second problem faced is the fact that raw sensor data hardly ever is useful as such to an application; instead it needs to be abstracted to turn it into a useful piece of information. Third, context needs to be combined from many distributed and heterogeneous sources, which in turn creates uncertainty that needs to be dealt with

by the application. Finally, to provide a positive user experience, an application must adjust its actions in real-time to continuous changes that take place in the context. These problems altogether have resulted in developers building new applications from scratch instead of reusing code or design ideas from previous applications. According to Dey and Häkkinen, toolkits and infrastructures aimed at addressing these issues have been attempted many times. The outstanding problems in these solutions include “representing and querying context using common ontology, algorithms for fusing heterogeneous context together, dealing with uncertainty, and inference techniques for deriving higher level forms of context such as human intent”.

Laakko & Gustafsson [27] elaborate the problems in understanding context. The reliable measurement or identification of the context is a major challenge related to context information processing. There are numerous low-level contextual aspects, such as physical, technical or social, which have to be taken into account when trying to understand what is happening at the moment, i.e., what is the user-level context. Zhang *et al.* [4] continue describing this challenge. When the context-sensing infrastructures for context-aware applications are distributed, low-quality or even erroneous outcomes may result. Alternatively, context ambiguity related to the difference between what the user expects and what the application, on the other hand, anticipates may also occur. Another challenge related to sensor-based context-aware mobile systems is the noise and ambiguity of sensor data. For this, two approaches can be considered: the more tedious one is to allow user to disambiguate the context data. The more feasible approach is to integrate data from multiple sensors to gain a better view of the overall situation.

Characteristically from a mobile usage perspective, the context-aware application use sessions are short, spanning from few seconds to few minutes, and it often involves some other activity alongside [49]. Consequently, information the user needs should be available quickly and effortlessly. According to Gustafsson [20], the issues that arise from using a mobile device user interface include the tediousness of the input of textual information and the small display, which is able to present only so much information at any one time. These challenges should be taken into account when designing a user-friendly context-aware mobile service.

3.4 How to design context-aware mobile services and applications

At this point, it is necessary to point out that the guidelines for designing context-aware mobile applications should also be kept in mind when designing context-aware mobile services. Always, when context-awareness is part of the design, the same issues need to be dealt with whether the information is processed in client- or in server-side.

In 2001, Dey [50] introduced the concept of a context widget, whose function is to hide the sensing of the context from the applications, which need that context information. Applications are able to access context information via widgets using polling and subscribing thus receiving the contexts they need to make further user-level context inferences. Distinctive in this paper was the fact that the widgets operate independently from the applications that use them. This eases the burden of the context-aware applica-

tion designer as the widgets need not be maintained from the application itself. The widgets may store previous contexts making it easier for the applications to predict the future actions or intentions of users. These contexts may be used in combination to predict a single piece of high-level context. Furthermore, the abstraction of *situation* is introduced: when relevant entities in the context-aware system are in a particular state, a particular situation has been invoked. The system complying with this may then be called *situation-aware*. This enables the application designers to concentrate on the core of the design process: what context-aware features their application should support and when they should be enacted.

Essentially, Dey and Häkkinen in [11] propose multiple design guidelines for mobile context-aware applications. 1) Select appropriate level of automation. This is a key issue as it affects the user interface, hence the whole user experience. The more there is uncertainty in context-recognition, the less the application should act proactively. The relation of the level of uncertainty in context recognition to the selected level of automation is depicted in Figure 7. 2) User control has to be ensured to maintain the feeling of being in control over the device. This can be implemented, e.g., with confirmation dialogues and with the ability to take control of the device and application at any time. 3) Avoid unnecessary interruptions. Distractions are undesired as they have impact on the user's performance and satisfaction with the system. 4) Avoid information overflow. It is only so much information a single user can fully focus on, so some kind of priority ordering needs to be defined. 5) Appropriate visibility level of system status. The user wants to be aware what the system is actually doing. For this, the visibility level of the system needs to be sufficient. 6) Personalization for individual needs. A context-aware application should respond sufficiently to individual needs. This can be realized, e.g., by filtering data according to user's preferences. 7) Secure user's privacy. Context sharing is one of the most relevant themes in privacy of the user. If not taken care of properly, trust, frequency of use and application acceptability may all be impacted negatively. 8) Take into account the impact of social context. In some social contexts, certain device or user behavior may be considered awkward or unacceptable. This calls for an appropriate balance of user-initiated and system-initiated actions. All in all, these eight guidelines need to be evaluated when the actual application or service has been implemented.

Zhang *et al.* [4] have identified a number of important issues that need to be explored and considered in the design process of context-aware applications. First of all, the design and implementation of such an application should be user-centered. This type of design process emphasizes the importance of understanding users in their natural use environment. Secondly, usability in interactions between humans and computers has not received as much attention in context-aware mobile application research as is necessary to create truly usable user interfaces. Academic researchers know how to design proof-of-concepts or functional demonstrators, so far without proper theoretical guidelines for narrowing the gap between UI design and technical development of context-aware applications. This needs to be addressed in the design process: empirical, user-centered approach to understanding contexts and their effect on users. Third, privacy of the users is a critical challenge in developing context-aware mobile applications. User acceptance is directly affected by issues such as ownership of location information and what kind of information is allowed to be collected by the application.

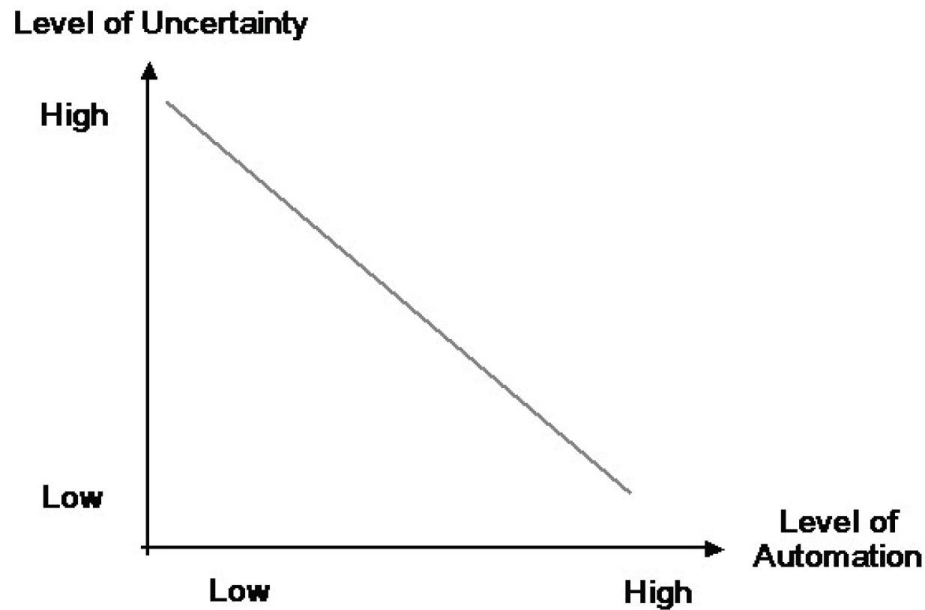


Figure 7. How the selected level of automation/proactivity should be chosen with regard to the level of uncertainty in context recognition. [11]

Fourth issue that needs to be considered is how much user interference is needed. For context-aware mobile application this means what level of automatic context-aware adaptation should be allowed. Too many interruptions from the application can frustrate a user. On the other hand, excessively automatic adaptation can also frustrate a user as she is forced to update her preferences constantly. The subjective nature of this problem makes it a very challenging issue. Fifth, information overloading is something that needs to be taken into account when designing context-aware mobile applications. A myriad of various sensors are collecting context data and interacting with the environment, which may result in a flood of, e.g., advertisements, local event notifications or other supplementary information. Machine learning may be utilized here to allow individual independent learning of context-aware preferences.

Similarly, Hiltunen *et al.* [5] discuss the methods for designing successful mobile applications. Conclusions follow the same principles as in [4]: understanding of users' perception related to the topic. Main issues are to define how a user distinguishes contextual information sources and their categorization, and the meaning and needs people propose for context-aware applications. Hiltunen *et al.* also investigated the views people have on defining and grouping context attributes. The results revealed strong inclination that people have very different perceptions of how to group context attributes and what the values or subgroups within a category should be. Categorizing of context attributes into a commonly agreed ontology proved to be very difficult, in addition to which the cultural or geographical dependencies also affected the categorization. This would then indicate the need for context attribute research per culture and country. Finally, the results showed that there are also attributes, which would be difficult for the user to set: those attributes can be measured, but hardly ever discussed in daily life.

Nummiaho [51] discusses the practical applications in context-aware communication. He identified the following goals to be important for different context-aware communication applications: 1) Right message at the right time, 2) Reminders only when they are actually relevant, 3) Sharing awareness of one's context and 4) Reducing ambiguity. 1) is about determining which people should be included in a communication based on context, which means improving relevance. 2) is related to delivering information when it is in the most timely and relevant context, i.e., minimizing disruption and reducing overload. 3) includes questions regarding communication partners and their context, whether or not the communication is appropriate at the moment. This helps people make intelligent communication decisions. The aim in 4) is to ensure that all communicating parties mean the same thing when they say something.

As the mobile users form heterogeneous groups, it is troublesome to design mobile services, which would be accepted by them all. To understand how and on what basis do these groups accept and adopt mobile services, various classifications of the user groups need to be established. In their paper, Zhang and Zheng [1] analyzed the types of end-users using mobile services. They divide mobile services into three categories: 1) one-to-one communication, 2) individual data service and 3) many-to-many communication. 1) refers to the services consumed between two individuals and are very essential and basic for every end user. The main concern for these users is the relation between the value and the cost. 2) provides a wireless access to an assortment of data services using a mobile device. Individual data services are driven by social needs of the business users, many times improving work efficiency and effectiveness. 3) is an emergent service that enables groups of people to communicate with each other. These services are "attractive", meant for trend-setters and technology enthusiasts, who are interested in, e.g., virtual communities or mobile gaming. The users of this category concern more about attractive features and behavioral requirements.

3.5 Content recommendation in context-aware DVB-H mobile service

From what was presented in Sections 3.3 and 3.4, it becomes clear that designing a context-aware service – let alone a context-aware *DVB-H* service – imposes a substantial amount of careful planning. Using *DVB-H* as the content delivery medium certainly does not ease the design process of context-aware *DVB-H* service. At minimum, it is now essential to take into account the guidelines presented above and make sure most of them become satisfied in the designed service, especially usability.

While at the moment it seems unfeasible to design a completely context-aware *DVB-H* service, it seems like a good idea to strive for it. In the transition phase – from the traditional, ordinary mobile TV with content push into a totally context-aware system with pulled content – it might be sensible to design something from in-between the two far-ends. One possibility would be to design a *DVB-H* service that, in traditional sense, pushes content according to a schedule, but is able to make recommendations of the content that is about to be broadcast. The recommendations would be based on comparing the metadata of the content with user's preferences and current context, which can be acquired from an external service. This way the user of this context-aware *DVB-H*

service could be informed of the content she might be interested in the near future, thus enabling efficient media consumption.

Assuming that the software framework described above was implemented, several challenges would emerge. One problem is information representation: how to describe content metadata and user context in such a way that they can be compared with each other in order to make reliable inferences of real needs of a user. Another issue is related to the actual inference process: what kind of conclusions can actually be drawn by comparing content and user context metadata. It seems as if only the limits of application designer's imagination are bounding the developing of more and more complicated rules, by which the inferences can be made.

The next Chapter establishes the background and requirements for designing a context-aware DVB-H service, and then proceeds on to presenting the Context-Aware DVB-H Service Software Framework developed in this Thesis. Further on in the 5th Chapter, the design is realized with a reference implementation for demonstrating the framework design is actually feasible, as opposed to plain theoretical speculation.

4 DEVELOPED SOFTWARE FRAMEWORK

The motivation for starting the development of Context-Aware DVB-H Service was related to CoDe-project [12]. CoDe is short for the project name “Situating Scalable Multimedia Content Management and Delivery for Heterogeneous Mobile Systems”. As the project name suggests, the objective of CoDe was to develop an integrated framework for mobile services, which are context-aware, scalable and utilize several delivery networks. The intelligence of the services developed in CoDe is aggregated by Orchestrator Service, which is ultimately responsible for understanding the situation of an individual user, i.e., what particular content would be relevant for the user right now. In practice, Orchestrator Service compiles so called information packages, which are composed of situationally relevant pieces of content. The developed client application would then receive these information packages and adapt the content and context furthermore according to the needs of user. Substantially, the architecture of CoDe framework is service-oriented, which suits perfectly the objective of developing Context-Aware DVB-H Service.

When CoDe-project had progressed for some time, it was perceived as necessary to develop a separate content recommendation service framework, which, in addition to being context-aware, would utilize DVB-H as the primary means of delivering content to users. The responsibility for developing and implementing this service was given to the author of this Thesis. CoDe framework consists of several separate services, but Context-Aware DVB-H Service was the main contribution by the author to CoDe-project, and it is also the motivation for writing this Thesis.

In this Chapter, the most important aspect is the detailed description of the developed Context-Aware DVB-H Service Software Framework from a theoretical perspective. However, the specific requirements for developing such a framework need to be defined first. This is done by examining a very simplistic use scenario, from which certain requirements can be identified. After the requirement specification, an overview of the framework is given in Section 4.2, which is followed by describing the ways of representing information between the various entities in the framework in Section 4.3. Sections 4.4 and 4.5 explain how the information required to make the recommendations is obtained within DVB-H Service. Sections 4.6 and 4.7 explain how the recommendation process proceeds once it is initialized. Section 4.8 examines the functionalities of the client application related to the framework.

4.1 Framework requirements

Basic scenario for specifying requirements

John is a user of Context-Aware DVB-H Service. According to his announced interests, John is interested in TV-series when he is at home. As DVB-H Service has knowledge of this fact, it sends a recommendation to John during the workday. This recommendation informs John that his favorite TV-series, Band of Brothers, is broadcast in the evening at 8 p.m.

In order to design a context-aware DVB-H service with all the required functionalities featured in the example scenario above, the background processes of the scenario need to be considered. From these processes, the requirements can be identified. Furthermore, it is important to include the requirements related to context-aware mobile services and applications from the previous Chapter to the design. Some architecture-related advice is also assumed from the earlier work in Section 2.6.

Considering the above scenario, first of all, DVB-H Service needs to know John is actually a subscriber of this service. For this, an entity for keeping track of the subscribers is needed in the designed framework. Secondly, John's preferences, and profile and context information need to be known by DVB-H Service. A way of collecting and maintaining this information is another resource needed by the software framework. Third entity needed is access to information related to the broadcast content. This way user's interests and context can be compared with the upcoming content broadcasts. Scheduling also needs to be carried out in a uniform way for DVB-H Service to be able to make timely recommendations – in this particular scenario it is during the workday. Now, when all the information related to each party involved is available, inferences regarding users' interests towards specific content can be initialized. For this, an inference engine within DVB-H Service is needed. This inference engine would be able to deduce that John will be interested in Band of Brothers, and recommend it to him. In general, the inference engine will be responsible for making the decisions about whether to recommend content to users of the service. An easy-to-use, efficient and fluent client application is needed for John to be able to view the recommendations in a user-friendly and effortless way. In the following, the requirements described above are further elaborated with the help of the earlier work from Section 2.6 and guidelines from the previous Chapter.

The whole notion of recommending content in the form of notifications can be seen advantageous, as it was also successfully applied in Notification Service for DVB-H Mobile Broadcast in Section 2.6.4. For the scope of this Thesis, it is enough to serve recommendations related to any DVB-H content, instead of the more specific service-related notifications depicted in Section 2.6.4. As in MyTV 2.0 (cf. Section 2.6.3), the recommendations in Context-Aware DVB-H Service should be delivered via 3G network instead of using DVB-H. This allows easier interaction with the client application.

When the need for Context-Aware DVB-H Service rose in CoDe-project, the most important requirement was to include the Delivery Context Service into the framework to be designed. Delivery Context Service was jointly developed earlier in CoDe-project to enable context-awareness and personalization in the CoDe framework. Since DVB-H Service also needed to incorporate these two properties, it was seen as relevant to include Delivery Context Service into the framework developed in this Thesis. This way, the integrated framework of CoDe could benefit both from Delivery Context Service and from the separately developed Context-Aware DVB-H Service, whose component Delivery Context Service at the same time is. The users of DVB-H Service comprise a subset of the users of the CoDe framework. It is assumed that the users of DVB-H Service have already been using some other CoDe-based service(s). This is how profile- and context metadata of the users is already included in the DVB-H Service framework

– they need not be collected separately. Delivery Context Service is considered as a 3rd party component in Context-Aware DVB-H Service. This decision follows the system architecture of FLAME2008 depicted in Section 2.6.2. Delivery Context Service will be described in more detail in Section 5.1.1.

For Context-Aware DVB-H Service to be able to offer DVB-H content, we need a content provider for delivering the content, or at least the content metadata, which indicates the location of the actual content. The content metadata needs to be precise and descriptive, since it is used for deciding whether the content is interesting to a user of DVB-H Service. This content provider should be a 3rd party component, because it is not at the core substance of the developed service. This was also the case in the architecture of both FLAME2008 and MyTV 2.0 in Sections 2.6.2 and 2.6.3, respectively. Necessary component required also in Context-Aware DVB-H Service is some service that keeps track of the subscribers of the service. Since Context-Aware DVB-H Service is part of the bigger framework of CoDe-project, User Information Service developed for CoDe is sufficient for the development of this service, again as a 3rd party component for the same reason as previously. The next component to consider is scheduling. Scheduling of content broadcast was perceived to be a task of the DVB-H Service administrator. Scheduling could also be done by an external entity, but in the scope of this Thesis, and considering the fact that CoDe-project had access to a DVB-H transmitter and a DVB-H scheduling software, this is seen as a sufficient way of conducting the scheduling.

The most generic principles for the actual recommendation process, which takes place in the core of Context-Aware DVB-H Service, can be adapted from FLAME2008. In FLAME2008, situation profiles, derived from user profiles and contexts, were matched against service profiles. In Context-Aware DVB-H Service, these service profiles can be translated as the metadata, which is composed of content, schedule and service information. This metadata can then be compared against the user profile and context metadata. Going into more details regarding the requirements of the recommendation process and the client application, it is relevant to take into account the design guidelines presented in the previous Chapter. In the following, these challenges and guidelines are considered when establishing the general requirements for the framework to be designed.

The client application is all that matters for the users of DVB-H Service. This is why one of the main requirements for it is to be informative, clear, have a fluent navigation and smooth user interaction. A proper level of automation must be considered, even though this property is related more to use habits of individual users than on any standard principles. In the scope of this Thesis, the level of automation will not be adjustable, although it will be kept on a “decent” level. Thus, user control will be fixed to a certain level as well. It is advisable not to forget the importance of the level of automation in relation to user control, since it affects the overall experience of the designed service. The application must give enough feedback to the user to ensure high usability standards. Considering privacy, it is perceived as an important factor with regard to user satisfaction. However, as designing Context-Aware DVB-H Service is mostly about demonstrating DVB-H can be used in a context-aware way, a decision was made not to

consider privacy in the designed framework. On the other hand, what was seen as relevant is to avoid information overload, which can be avoided in server-side of DVB-H Service by, e.g., minimizing redundant notifications or prohibiting notifications in certain contexts.

The uncertainty in context recognition is a problem, for which there is no standard solution. This problem becomes evident at latest in the client application in the form of wrong notifications. The designed framework is highly experimental with next to none previous work with guidelines to assist on how to recognize contexts reliably. Thus, the requirement for context-recognition is, first of all, to develop an inference system, which refines further the context and profile information served by the Delivery Context Service component. The validity of the context recognition can then be ensured by implementation scenarios diverse enough to assure it works in practical situations. Another issue that needs special attention is how to represent information in the designed framework. The requirement here can be defined as such that the information representation should allow clear and effortless way of comparing metadata. This way the inference process will also become easier. At this point, it was seen ontologies are too heavy and extensive for the purposes of this Thesis, so a decision to use XML Schemas for information representations was made. However, the problem with using XML Schemas is the lack of standardization and thus poor interoperability. Then, another requirement here is to design the Schemas as interoperable as possible. One way of doing this is to investigate the terminology used in DVB-H ESG Schemas, which offer relevant vocabulary for, e.g., content metadata.

The framework to be designed is quite generic, so the user groups for the service need not be specified particularly. It is sufficient to say the users of Context-Aware DVB-H Service are those users, who use CoDe services as well and are interested in receiving content via DVB-H. These users, in general, are technology enthusiasts, who are interested in attractive features and behavioral requirements of the service, as stated in the previous Chapter. However, the framework design should allow its modifying in a way that allows varying and diverse user groups to adopt the service, even though this is in no way an essential issue here.

Note that the requirements elaborated above are the fundamental requirements. These requirements will be verified by examining the performance of more intricate scenarios of Section 5.3. In the following Sections, Context-Aware DVB-H Service Software Framework is described in detail from a theoretical perspective. Overview of the framework is given first.

4.2 General view of Context-Aware DVB-H Service Software Framework

To enable the users of a DVB-H-based service to acquire information regarding content, which is interesting and relevant, *Context-Aware DVB-H Service* was designed and implemented. For the users of this *software framework*, Context-Aware DVB-H Service offers relevant recommendations of content soon to be broadcast. The user may then utilize this information according to her needs in that particular situation, time, place and so forth. The service itself consists of *DVB-H Recommendation Service (DRS)* and

DVB-H Client (DC). The software framework, including the previous two entities, is complemented with external services, which are essential for the proper operation of the design. The communication between DRS and DC, related to the recommendations, takes place in 3G-network. The actual content is delivered via DVB-H network.

Context-Aware DVB-H Service sets itself somewhere in between the traditional push-based digital TV-service and fully context-aware DVB-H service, which would offer the user the content she needs and wants at any given situation without any notifications. In a conceptual sense, Context-Aware DVB-H Service has inclination towards pull-based service, as the user is made aware what she “wants” to see. The content itself, however, is pushed traditionally to the user, as the case in a broadcast network inherently is. Context-Aware DVB-H Service is actually a separate service in relation to DVB-H, i.e., it does not need to operate in a DVB-H service operator’s premises. Still, the use of DVB-H is a requirement for this software framework to function correctly, since it is designed specifically for DVB-H type of content.

Overview of Context-Aware DVB-H Service Software Framework is presented in Figure 8. Substantially, DVB-H Service consists of DRS and DC. These two entities were the main components developed for this framework. The former is responsible for generating the actual recommendations server-side, while the latter is operated by the user with her mobile device to fetch and view the recommendations. In addition, the complete framework that was developed includes a *Content Service Provider*, a *User Information Service* and a *Delivery Context Service*. In this framework, user is the content consumer via her mobile device, which contains DC. Moreover, user is responsible for inputting and editing the profile information and her preferences to Delivery Context Service.

Content Service Provider is composed of Content Provider, which is responsible for delivering content metadata to DRS. The actual content sources are also considered as a part of the Content Service Provider, although they may be completely separate services in relation to the Content Provider. This is irrelevant for DRS, as content metadata always embodies the content source information, i.e., information indicating where the content is located in the network. The DVB-H transmitter in Figure 8 is operated by DRS; this is not necessary as the DVB-H network can also be owned by an external network operator. Depending on the ownership of the DVB-H infrastructure, the role of Content Provider changes. If DRS is responsible for the actual broadcasting of content and decides what to broadcast, the Content Provider acts as a pull-service: DRS requests metadata for the content it has scheduled for broadcasting. On the other hand, if the DVB-H infrastructure is operated and broadcasts decided by an external operator, DRS functions as a plain recommendation service, which needs both content and scheduling metadata in order to make the recommendations. In this case the Content Provider pushes content metadata to DRS, e.g., on a weekly basis.

User Information Service is part of the CoDe-project. The CoDe-service users’ login and other related data is stored in User Information Service database. As User Information Service is as well a part of the framework developed in this Thesis, the relevant parts of the user information are also utilized in DVB-H Service. Likewise, Delivery

Context Service is a component in the CoDe-project. This service can still be decomposed into its two principal components – *Context* and *Profile Service*. The former is responsible for collecting and inferring contexts of the users of the service, and for serving the clients requiring this information. The latter accounts for collecting, storing and serving the profile information related to the users of the service.

All metadata, including content, user profile, user context, schedule and service metadata, is represented by using markup language specified with XML Schemas. Ideally, an ontology-based model would be utilized. However, this is not in the scope of this Thesis. DVB-H Service uses XML Schemas to specify the format, in which information is meant to be represented. This means a markup scheme model (cf. Section 2.3) is utilized in expressing metadata. Assumptions need to be made in order for this scheme to be useful in the present software framework. The utilized XML Schemas and the vocabularies used for describing them with XML-documents need to be well-known by all parties involved. This means all parameters and contexts and such are agreed upon. Should there be any changes, all parties must be informed appropriately.

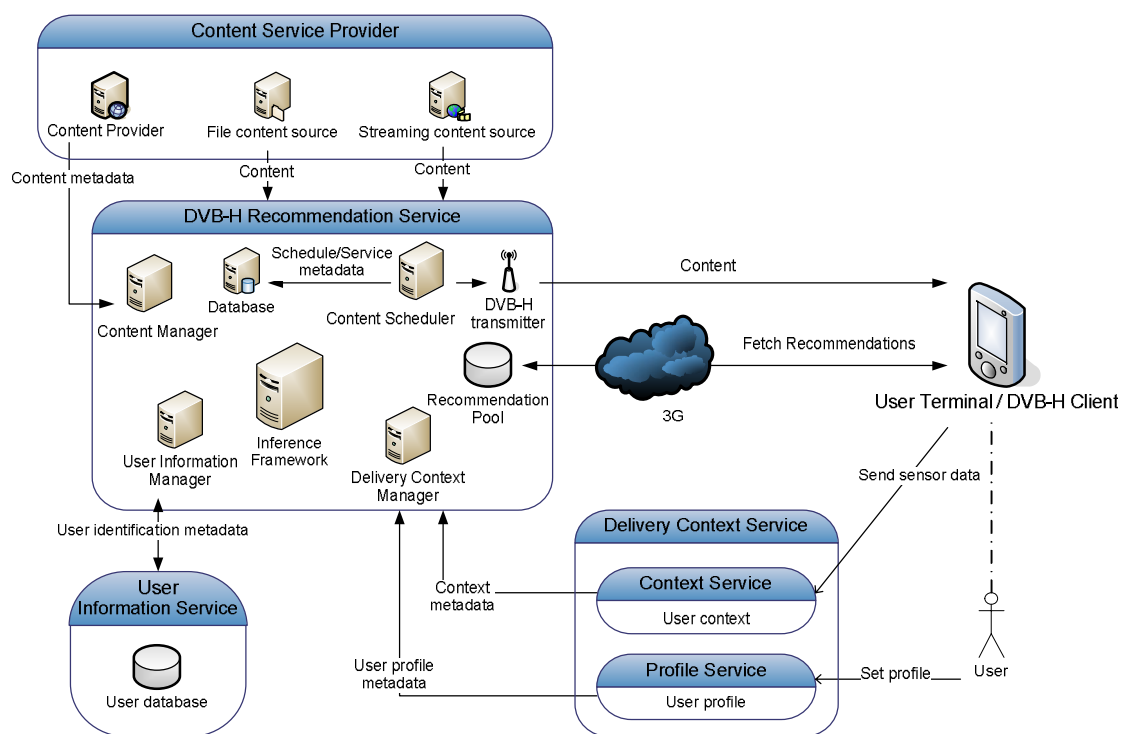


Figure 8. Overview of the Context-Aware DVB-H Service Software Framework. Content Service Provider provides at least content metadata. It may provide the actual content itself as well. User Information Service keeps track of the users of the service. Delivery Context Service communicates with user terminals for obtaining the latest context and profile information. On request it sends this information to DVB-H Recommendation Service. User Terminal is operated by the user via DVB-H Client application, whose main function is to fetch recommendations via 3G network from DVB-H Recommendation Service. Most importantly, DVB-H Recommendation Service is responsible for receiving and aggregating various metadata, and utilizes these to make relevant recommendations for the users of DVB-H Service.

4.3 Metadata descriptions

To enable the communication between different entities in Context-Aware DVB-H Service Software Framework and within DRS itself, there is a need for describing information coherently. In this manner all the parties are able to send and receive information in uniform and understandable way. For this, XML Schemas are utilized.

There are seven different types of XML Schemas used in Context-Aware DVB-H Service:

- 1) *DeliveryContextSchema* comprises two parts: *ContextSchema* and *ProfileSchema*, which both describe relevant user metadata regarding her situation, preferences and technical parameters related to the user terminal. The metadata complying with *DeliveryContextSchema* is used for communication between Delivery Context Service and DRS.
- 2) *ContentSchema*, according to which Content Provider assembles the metadata describing the actual content. In addition to describing the actual content, *ContentSchema* also defines what kind of technical specifications are required in order to consume the content properly. Context-wise the most important information in *ContentSchema* are the tags that describe in what kind of situations and with what preferences the content is most likely to be consumed. Content Provider delivers this metadata to DRS.
- Equally important are 3) *ScheduleSchema* and 4) *ServiceSchema*, which describe scheduling and service, or channel, information. *ScheduleSchema* describes when the content is broadcast, what is the duration of the broadcast and identifies the content item related to this particular schedule item. The most important function of *ServiceSchema* is to describe the multicast IP address and port of the particular service offering content.
- The two previous XML Schemas are based on XML-data output from a generic DVB-H scheduling software. For the output, an XML Schema, describing the form and vocabulary of the output, needs to be defined. Because the used scheduling software is decided per implementation, it cannot be given any specific name. For the time being, let this Schema be called 5) *SoftwareSchema*. Its name shall be later on redefined as determined by the name of the used DVB-H scheduling software in the reference implementation.
- 6) *RecommendationSchema* describes information that is relevant for the end-user to see in order to be able to decide whether or not she is interested in the particular piece of content. An XML-description based on *RecommendationSchema* is composed along the inference process by DRS. This is the XML-description DC fetches from DRS to receive new recommendations.

- 7) *UserInformationSchema* is used in communication between User Information Service and DRS. This Schema describes in a very simple manner the user identification information: user name, device ID (e.g., IMEI-code of mobile device) and whether or not the user has an active session ongoing. The latter information is needed to make more accurate recommendations.

4.4 Obtaining user information, profile and context

To be able to offer recommendations for every user of DVB-H Service it is relevant to know who they are. For this purpose, User Information Service is used: it keeps track of the users signed up for this service. User name, device ID and whether the user has an active session currently are accounted for. This information is delivered on a regular basis to DRS. To be more specific, the information is delivered to *User Information Manager* in DRS, also visible in Figure 8. It is responsible for receiving and processing the user data. The data is fetched on regular intervals. Using this data DRS is capable of querying for the contexts and profiles of specific users and prepare recommendations for them. This information is processed further into DRS database. A single user information item could, for example, be as follows:

```
<user>
  <username>John</username>
  <deviceId>935827253978</deviceId>
  <hasSession>>true</hasSession>
</user>.
```

Delivery Context Manager in DRS, also visible in Figure 8, reacts to the requests by other components in DRS. Its main functions are to request profile and context metadata information of individual users of DVB-H Service from Delivery Context Service, and to compose a metadata description of each user by generating an XML-description complying with *DeliveryContextSchema*. This description is delivered in response to any entity requesting profile and/or context information.

User profile metadata is composed of two parts: *general* and *active*. General profile is information related to the most basic and static data regarding the user, such as name, age and so on. Active profile is the specific profile type that the user has been detected to be using last. The most relevant information here are *interests* (and also *disinterests*) of the user. User profile metadata is identified with a “user name – device ID” -pair.

User context metadata is identified exactly in the same way as user profile metadata. Context information describes where the user is located and what her active contexts are. Additionally, the support for various content formats, the technical parameters of mobile device and the user calendar information are described in the metadata. The active context information is utilized in understanding the situation of the user. This inference is carried out by DRS *Inference Framework*, visible in Figure 8.

4.5 Acquisition of content metadata and content scheduling

Content Provider sends metadata related to the content that is soon to be broadcast. This metadata is sent in an agreed format specified by ContentSchema. It includes information related not only to the content (e.g., title, duration, subtitle languages, genre, etc.) itself but also technical parameters required for the proper consumption of the content with a mobile device (e.g., MIME-types, resolution...). In addition to the content and technical information, substantial information is the keywords that can be used to describe any kind of references essential to the content. A special type of keyword is a *locationKeyword*, which refers either to the whereabouts of the events of the content or to the locations where the content is meant to be consumed. With this information better inferences can be made. Finally, the network location of the content is described in the content metadata. *Content Manager* in DRS receives the above-described metadata and processes it onwards to DRS database. In the database, each piece of content, or content metadata item, is given a content ID, with which it can be identified.

Scheduling information is processed by DRS *Schedule Manager*. It receives scheduling metadata from a DVB-H scheduling software used in DVB-H Service. The received metadata thus complies with SoftwareSchema, i.e., the Schema from the output of the implementation-specific DVB-H scheduling software. From this XML-metadata Schedule Manager generates two XML-descriptions: schedule and service. Schedule metadata describes in detail the starting and ending times for the content broadcast, the duration of the broadcast, the content and the service IDs that are related to this specific schedule item and, if applicable, data carousel technical data (IP, port). Data carousel is used in DVB-H for content download services. Service metadata deals with the information relevant for broadcasting: IP, port, service bandwidth, service type and so on. When the Schedule- and Service-XML-documents are generated, they are ingested into the database. The identifiers – schedule ID and service ID – need not be assigned in the database as they are already given in the XML-document complying with SoftwareSchema.

4.6 The recommendation inference process

From here on in this Thesis, an entity, which is the combination of inter-related content, schedule and service items, shall be simply called a *schedule item*. This is because a schedule item holds the information, and thus the access, related to the other two pieces of information items. In short, a schedule item refers to an entity, which depicts what, when and how something will be broadcast. Substantially, Schedule-XML used in DRS functions as an aggregator of server-side metadata: it combines schedule, content and service metadata together via the use of IDs referring to correct metadata items.

Next, the recommendation generation process is described in detail. It is elaborated from a functional perspective to give the reader a clear view of the process in the order of events occurring, i.e., how the recommendation generation begins and where it ends. The descriptions in this Section can be at all times compared against the diagram in Figure 9. Phases 1 and 2 in Figure 9 are illustrated with example timestamps, which represent at what times three consecutive inspection processes would start.

In the most typical case, DRS entity called *Schedule Inspection Manager* initiates the process for determining whether any user of DVB-H Service is interested in receiving a piece of DVB-H content. There are two different mechanisms for starting the process: schedule-based and calendar-based. In schedule-based mechanism, Schedule Inspection Manager polls DRS database for schedule items in one minute intervals. Only those schedule items, which are set to be broadcast within 60 minutes, are retrieved. From this subset of schedule items only those, which are set to be broadcast exactly either in 60 minutes or 5 minutes, are further processed. The time values mentioned are determined by the author – they are susceptible to modifications. The schedule IDs and the recommendation types, namely *60m* or *5m*, related to these items are sent to DRS *Inference Engine* for processing.

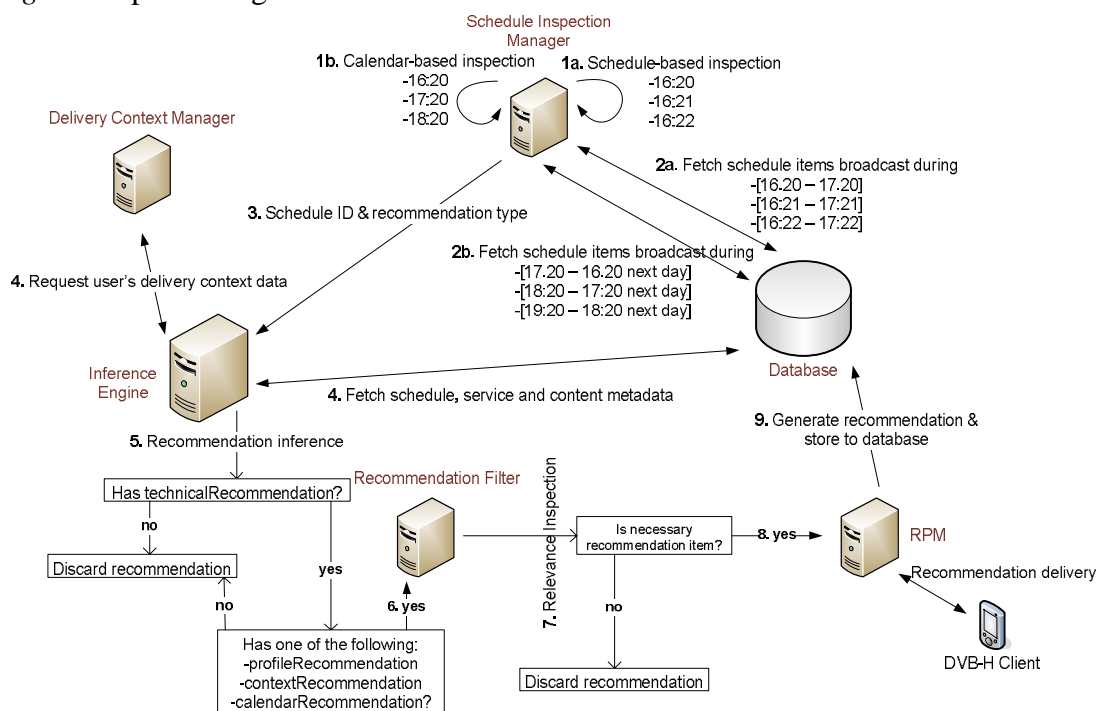


Figure 9. Recommendation generation process in DVB-H Recommendation Service. Schedule Inspection Manager performs either 1a or 1b and fetches schedule items accordingly (2a and 2b, respectively). The relevant schedule IDs and recommendation types are sent to Inference Engine (3), which both requests DVB-H Service users' delivery context data and fetches other relevant metadata (4). Using these two entities, inferences are made as to decide whether the content is suitable for a particular user (5). The recommendation item proceeds via Recommendation Filter (6, 7) to Recommendation Pool Manager (8), which finally generates the recommendation and stores it into a database (9).

Calendar-based mechanism is related to the schedule items in the time interval [current time + 1 hour, current time + 24 hours]. All schedule items within this time interval are relevant, but the mechanism itself is triggered only once in an hour. The schedule items here are utilized later on in searching the calendars of the users of DVB-H Service for entries that might be related to the relevant schedule items. A related calendar entry must have a starting time *after* the broadcast of the schedule item but within 24 hours from the current time. This is because the content should be available when the calendar

event occurs. Again, the schedule IDs and the recommendation type, namely *24h*, related to the relevant schedule items are forwarded to DRS Inference Engine.

On a high abstraction level, Inference Framework shown in Figure 8 consists of *Inference Engine*, *Recommendation Filter* and *Recommendation Pool Manager (RPM)*. They are responsible for creating, modifying and delivering the recommendations for the users of DVB-H Service.

4.6.1 Inference Engine

Inference Engine is responsible for deciding whether a particular schedule item can be recommended to a user of DVB-H Service. In general case, Inference Engine receives relevant schedule ID(s) and recommendation type(s) from Schedule Inspection Manager, once every minute. Based on the schedule ID, it then fetches the Schedule-, Service- and Content-XML-descriptions related to the particular schedule item from DRS database. At the same time, a request to fetch every user's delivery contexts (profile and context information) is sent to Delivery Context Manager, which subsequently returns an XML-description complying with *DeliveryContextSchema* containing the data from all the users.

Now Inference Engine has all the metadata it needs to decide whether a schedule item should be recommended to a user. One schedule item at a time all users in the *DeliveryContext-XML* are processed. Simply put, every piece of content coming into broadcast in the near future is being checked whether or not it would be interesting to any user of DVB-H Service. Next, relevant data from the schedule item is extracted and compared against user's delivery context in essential parts. This comparison, in general, is done by comparing various XML-elements in the schedule item to the "counterpart" elements in user's *DeliveryContext-XML*. These comparisons are performed for every user of the service. The comparisons are described next in more detail.

For DRS to be aware of the reasons why a recommendation was or was not given for a user, recommendations are intermediately separated into four categories: *technicalRecommendation*, *profileRecommendation*, *contextRecommendation* and *calendarRecommendation*. After the whole recommendation process, a user has none, some or all of these recommendations that are related to the schedule item at hand.

The first operation in the comparison process is investigating whether there are any major technical impediments – mainly client device -based – preventing the consumption of the particular content. These factors might, e.g., be the inability to display a specific video or the lack of available space on the device. Based on this comparison, a decision regarding *technicalRecommendation* is carried out. A *technicalRecommendation* is required for the recommendation process to proceed for a particular user.

After the decision regarding *technicalRecommendation*, user profile and context information is compared against the schedule item metadata. The comparisons are made with specific rules, which are given a certain amount of points. These points are used to

decide whether the particular schedule item should be recommended to user. When a matching pair of data is found, it is given some predefined amount of points. If, at the end of comparing the schedule item metadata (content, schedule and service metadata) to user's delivery context (profile and context metadata), a fixed threshold value for the amount of points is exceeded, then depending on the matching values, profile- and/or contextRecommendation may be given to user. However, contextRecommendation can not be given unless the user has an active session ongoing – otherwise it would not be sensible: context of the user might have changed. Likewise, if the piece of content related to the schedule item requires specific contexts to be present but the user does not have them, contextRecommendation can not be given. The process described above applies only to recommendation types 60m and 5m.

In an alternative case, when the recommendation type is 24h, i.e., calendar-based, after technicalRecommendation has been resolved, only user's calendar entries are considered from the delivery context. The calendar entries of the user are examined one by one and compared against the broadcast starting time of the current schedule item. If, during the time interval from current moment until the time the currently examined calendar event is supposed to take place, there is interesting content to be broadcast, then a calendarRecommendation will be given to this recommendation item. Interesting content here means some content the user might be interested according to her calendar metadata.

If a schedule item for a specific user has a technicalRecommendation and any of the following – profile-, context- or calendarRecommendation – it becomes a recommendation item. The recommendation item is unique as it includes a schedule item, which defines a specific piece of content that is scheduled to be broadcast on a specific time and is intended for a specific user. This recommendation item is sent forward to Recommendation Filter.

4.6.2 Recommendation Filter

The main function of Recommendation Filter is to make decisions on whether the earlier user recommendation items related to a particular schedule item should be replaced with newer ones. For example, a 5m-type recommendation item might replace an existing 60m-type recommendation item in the database in certain situations. This type of filtering is needed for minimizing the amount of unnecessary user interruptions, for there is inherently the possibility of creating multiple recommendations related to the same schedule item and user. For example, if a user has already fetched a recommendation, she most likely does not need the same recommendation again, other than maybe as a reminder.

Generally speaking, the earlier recommendations related to the same user and schedule item are replaced with newer ones, if the earlier ones have not been yet fetched by the user. If the earlier recommendation has already been fetched, the newer one is discarded as it is no more needed. However, an exception to the rule does exist: if some piece of content that is scheduled to be broadcast in 60 minutes has been recommended

to the user, but is no longer relevant to the user at the time, when the broadcast is 5 minutes away, the recommendation will be forwarded. It is called an *unrecommendation* as it would inform, if necessary, the user of the content not being anymore relevant to her. Recommendations passing the Recommendation Filter are forwarded to Recommendation Pool Manager.

4.6.3 Recommendation Pool Manager

Recommendation Pool Manager (*RPM*) is responsible for two separate tasks: generation of Recommendation-XML and delivering recommendations to users on request. In addition related to RPM, the *Recommendation Pool*, which in practice is DRS database table containing the recommendations, has a timer-based functionality, which is responsible for deleting expired recommendations. Expired recommendations are those recommendations, whose referred content has already been broadcast. These are cleared from the Recommendation Pool, which, in practice, is a database table containing the recommendations.

The generation of Recommendation-XML metadata complies with RecommendationSchema. It is composed based on the recommendation item from Recommendation Filter. This metadata consists of pieces of technical, content, schedule, service and other metadata. The latter metadata can be, e.g., warnings related to the mobile device parameters, or it can be high-level context information needed for deciding if the recommendation should be delivered to the user on request. The recommendations are finally stored into Recommendation Pool, which assigns the recommendation an ID.

If DVB-H Client makes a query for new recommendations, RPM checks if there are any new recommendations for the user. If there are, RPM inspects whether the recommendation can really be sent, i.e., at this point the *user must have all the required contexts present and the do-not-disturb -context must not be present*, as indicated by the recommendation. If everything checks out fine, the recommendation is sent to the user. However, if the requested recommendation does not pass the abovementioned requirements, it is sent back to Inference Engine along with the necessary user information. This is done, because the situation of the user might have changed in the time interval, when the recommendation was made and when the recommendation was requested by the client. This type of recommendation is separated from the ordinary 5m, 60m or 24h -type recommendations by labeling it as an *instant* recommendation. Inference Engine now checks whether the requirements mentioned above are satisfied and informs RPM accordingly.

4.7 Client application

The DVB-H Client application is started by the user. It then runs as a background process, which sends queries in certain intervals to DVB-H Recommendation Service for new recommendations. If there are new recommendations available, they are received, parsed and finally presented to the user in a Graphical User Interface (GUI). If there are

no new recommendations, the GUI only informs user that the DVB-H Service is active. User can exit or just hide the application. Hiding does not interrupt the background process. In the following, the background process of DVB-H Client is described in more detail.

- The client application polls DVB-H Recommendation Service once every two minutes if there are new recommendations available. The query contains the user's username and device ID, which are stored in the device memory. After the query, the polling process sleeps for two minutes.
- If the response from DVB-H Recommendation Service contains new recommendations, the recommendations are sent for processing, namely parsing. The parsing extracts all the relevant data from the recommendations. This data is such that it is relevant for the user to see in the user interface.
- After relevant data has been extracted, the GUI is built run-time from the data. This interface is then presented to the user via an alert, which notifies the user of new recommendation(s).
- User may browse through the recommendations that are bundled in the same fetch-instance. User may hide the recommendation(s) or exit – in both cases the background process continues uninterrupted.

4.8 Summary of the developed software framework

Context-Aware DVB-H Service Software Framework consists of DVB-H Recommendation Service, DVB-H Client, Content Service Provider, User Information Service and User Context Service. The first two entities are the most important ones, even though without the latter they could not function properly. Together these five entities, along with the user herself, operate to provide context-awareness into DVB-H-mediated content, of which the user is interested of.

The developed framework is, in fact and to some extent, independent in relation to the actual DVB-H technology since the delivery network need not be DVB-H in particular. However, the framework is tailored for the needs of DVB-H content metadata and the specific ways of delivering its content. Communication between different entities in the framework is done simple by utilizing XML Schemas. The framework utilizes external services for user contexts and profiles, content metadata and the service subscribers. Scheduling is done with a DVB-H scheduling software within the service itself. The process, i.e., the need for recommending content is considered relatively frequently, which guarantees everything possible will potentially be recommended to the users of DVB-H Service. Recommendation inference is based on recommendation types: 5m, 60m, 24h and instant. They reflect how the recommendation should be handled and what special functions need to be considered for the specific type. The names of the first two recommendation types are related to the amount of time left before the particular content will be broadcast. "24h" refers to calendar-based recommendation

and “instant” means the recommendation validity should be decided immediately. The actual inferences done in Inference Engine are heuristic as they are not based on any specific, “standardized” rules or statistics of deciding the relevancy of content – mainly because there are none existing ones available. The recommendation relevancy for a specific user is decided with the help of a system of points, which reflect how interesting a particular piece of content is to the user. Recommendation Filter helps to minimize in any way redundant recommendations. Recommendation Pool Manager generates the Recommendation-XML served for the users and handles all the communications with the framework DVB-H Clients. DVB-H Client software mainly checks if there are any new recommendations and displays them for the user of the mobile device.

5 REFERENCE IMPLEMENTATION

Context-Aware DVB-H Service Software Framework was implemented based on the framework architecture and its defining requirements, which were elaborated in the previous Chapter. The functionality of the reference implementation was verified with the use of scenarios, which reflect various properties of the design requirements.

This Chapter first describes what technologies were seen as relevant and necessary to actualize the developed framework. Two more software components, highly specific to this implementation, are also presented in more depth. These two technologies are responsible for implementing Delivery Context Service and DVB-H Service scheduling software mentioned in the previous Chapter. The next Section then demonstrates how these all these technologies were integrated into Context-Aware DVB-H Service reference implementation in practice. Alongside, the various entities comprising DVB-H Recommendation Service and DVB-H Client and their information flows are explained. The following Section describes the scenarios, and explains what the actual processes behind the implementation logic are. Each scenario is described with explanations for what happens in each scenario from the technical perspective. The results from testing these scenarios are then used to evaluate the practical value of Context-Aware DVB-H Service reference implementation, as reflected by the initial requirements of the framework.

5.1 Implementation-specific background technologies

The technologies described in Section 2.7 were chosen for actualizing the functionalities of the reference implementation. They were considered as technologies simple, robust and well-known enough for the purposes of this Thesis's scope. The only major drawback in using Java ME is that currently there is no way of implementing the DVB-H download features into the client application. The other technologies suit well the requirements defined in the previous Chapter.

In addition to using the abovementioned technologies, two software components were needed to fully implement the desired functionalities. The first software component is responsible for actualizing Delivery Context Service of the developed framework. It is composed of two parts, User Profile and Context Service. The other component serves the purpose of DVB-H scheduling. This software is used to ingest scheduling and channel information, which then can be output for the use of DVB-H Recommendation Service. These software components are described here in more detail:

- *Delivery Context Service* is used for generating and collecting information related to the users and their preferences, etc. This is also the actual implementation of the Delivery Context Service of CoDe-project described in Chapter 4.
- *Sofia Digital Backstage Service Manager* for scheduling DVB-H content and extracting the metadata complying with SoftwareSchema. This Schema shall be henceforth called *SofiaDigitalSchema*.

5.1.1 Delivery Context Service

As part of the CoDe-project, a User Profile Service was implemented to enable personalization as part of the project’s integrated framework. This service is also utilized as the Profile Service component in the Context Delivery Service entity of the reference implementation described in this Thesis. Furthermore, a framework for context-aware applications, known as the Context Service, was designed and implemented in the CoDe-project by Gustafsson [20]. This service is also utilized as the Context Service component in the Context Delivery Service entity of the reference implementation. First, the original User Profile Service is shortly introduced. Following this, Context Service by Gustafsson is considered.

User Profile Service in CoDe-project is a framework used for creating, collecting and defining user profile information for various situations. The framework consists of Java Servlet server-side implementation and a client for entering the profile information. The user of this service enters static information into a general profile, which encompasses the most basic information regarding the user – the most important being user’s interests. Additionally, some pre-defined profile templates, such as “home”, “work” or “car”, are realized. User may define her own profiles with a “custom” profile template. The user profile information is stored server-side into MySQL database in XML-format. The class hierarchy and inheritance in the profile service is depicted in Figure 10. All profiles apart from the general profile can be used as the active profile of a user. The active profile changes according to the context of a user.

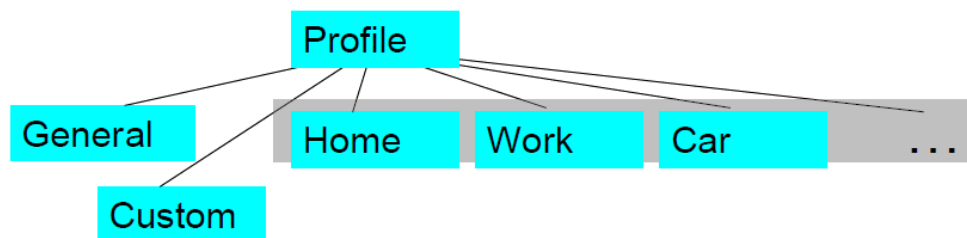


Figure 10. User Profile Service class hierarchy. General profile defines the basic static information, while various other profile types cover more specific needs and interests of a user. [27]

For the purposes of this Thesis, two essential components of the context-aware application framework are considered: context data gatherer and context server. Context data gatherer works as a background process on the mobile device. It collects low-level context data from the sensors of the device (Figure 11, phase 1), pre-processes the data and forwards the context update to the Context server (Figure 11, phase 2). In between the context data gatherer (“Sensor server” in Figure 11) and Context server lays a Context-aware client, via which all the context data is mediated. It is not considered here.

Context server is a centralized server for all mobile terminals. It utilizes the low-level context data collected from the context data gatherers, resolves a list of active user contexts (Figure 11, phase 3). In the scope of this Thesis, the 4th phase in Figure 11 is irrelevant, as updated context information is needed only server-side in DVB-H Service.

The Context server also collects context history and provides API for client applications for querying low-level or user-context data.

The context data gatherer collects, e.g., the following low-level data:

- Available wireless access points
- Used cell ID
- GPS data
- Accelerometer
- Connected accessories
- Phone calendar information

The Context server gets most of the low-level context data indirectly from the mobile terminal's Sensor server component. The Context server can also use low-level context data from external context sources, such as different Web services.

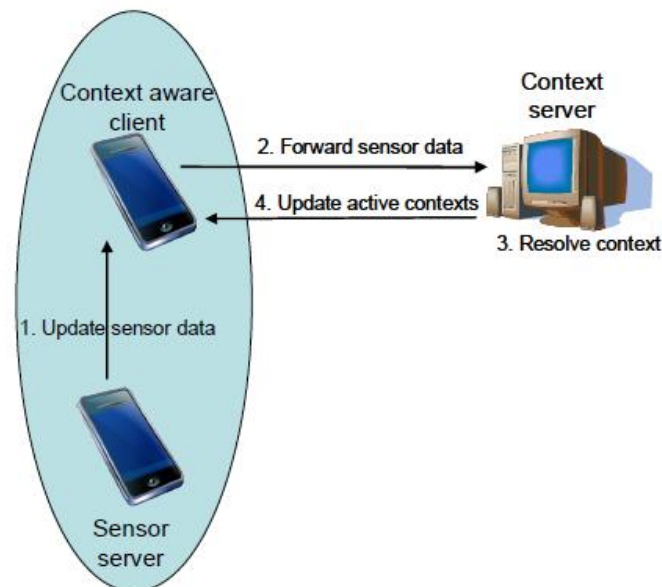


Figure 11. The main components of the context-aware application framework and high-level context inference chain in CoDe. The client holds a sensor server, which receives data from individual sensors of the device. Updated sensor data is then sent to a client application, which utilizes this information and sends it forward to an external context server. Higher-level contexts are resolved server-side. These contexts can then be sent anywhere, namely in DVB-H Service to Delivery Context Manager. [20]

Individual low-level contexts by themselves are rarely able to describe any situation. Combining low-level contexts then becomes relevant, which, in the case of the Context server, is called user-context resolving. As the Context server receives new context data from the context data gatherer, scoring is done based on certain rules. These rules define whether a certain user context is (most likely) active. If the active user context changes, an HTTP response is sent back to the application/service. The describing of the contexts is done with an XML-based language.

5.1.2 DVB-H scheduling using Sofia Digital Backstage Service Manager

Sofia Digital Backstage Service Manager (SDBSM) for DVB-H [52] is utilized for content scheduling and broadcasting in DVB-H Recommendation Service reference implementation. It is a solution for managing interactive services and content from various content sources on mobile TV networks. The Service Manager is operated with a Web-based UI, which is used, e.g., for scheduling content, managing service parameters and monitoring DVB-H transmissions. When a content item is about to be broadcast, SDBSM automatically initializes the procedure for content broadcasting on a specific service, or channel. These properties are utilized also in DRS. After the content scheduling is decided upon, the schedule information is entered into the Backstage Manager in an agreed way. Afterwards an XML-based schedule- and service description can be exported from SDBSM for the use of DRS scheduling.

For the purposes of Context-Aware DVB-H Service reference implementation, an XML Schema describing the exportable schedule- and service data was elaborated: SofiaDigitalSchema delineates the form of the metadata, which is exported from the SDBSM when the scheduling information has been entered. Using this metadata, DRS is able to compose two separate XML-descriptions representing schedule- and service metadata.

DRS implementation utilizes Sofia Digital Backstage Service Manager for actualizing the scheduling and broadcasting of the content. As an example of the SDBSM management interface, Figure 12 a) illustrates the configuration parameters of an example service. The most substantial pieces of information are service source, service input IP/port and bitrate. Figure 12 b) shows the page, where the scheduling of content is actually done. The contents of the parameter “Description (swe)” are related to information DRS needs in order to function correctly. “Title (eng)” contains the name of the content item and “Description (eng)” contains the content ID of the particular content item. Note that the scheduling in SDBSM can not be carried out until content metadata has been ingested into DRS. This is due to the fact that the MySQL database used in DRS is responsible for generating the content IDs. Thus, the content IDs must be extracted from the database and entered into the parameter field of “Description (eng)”.

5.2 Integrated reference implementation

Context-Aware DVB-H Service was implemented by utilizing the technologies described in Section 5.1. The way they are used in practice is portrayed in Figure 13. All the Servlets communicate with both each other and the external services by using HTTP REST. The MySQL database is accessed from the Servlets with MySQL Connector/J driver. DVB-H Client is implemented with Java ME. For the interested reader, the hierarchical relationships of the individual Java classes used in the implementation can be found from appendices 1 and 2 – for DVB-H Recommendation Service and for DVB-H Client, respectively. Appendix 1 shows how there are both Java Servlet-type classes and timer-based classes. The latter classes are responsible for operations based on fixed intervals. Appendix 2 shows the simplicity of the Java ME class hierarchy.

SOFIA BACKSTAGE SERVICE MANAGER
Servers / DVB-H / Broadcast-testikanava

Servers

- Sofia DVB-H
 - dummy server
 - DVB-H**
 - Social Video Trial
- GStreamer
 - Streamer
- Streaming
 - Video Streamer
- TS File Player
 - TS Player

System management
Users
Configuration

Library
Applications
Files
Data Sets
Videos
SDP Templates
Transport Streams

Scheduling
Publish
Calendar
Scheduler

admin

Edit Service 'Broadcast-testikanava'

Name: Broadcast-testikanava

Label: 6 Number: 6 ?

Enabled

Sort order [0 - 999]: 6

Broadcast and Unicast Service
 Broadcast Service
 Unicast Service
 Filecast Service

Service Source: niina_online_englanti_h.264.mov Apply

Service input IP: 232.10.1.14

Bitrate kbit/s [0 - 2147483647]: 305

Service input port [0 - 65535]: 10500

Figure 12 a)

SERVICE MANAGER
Calendar / Task

Calendar this week Today The server time is **2010-10-15 16:04:05**

Service: Broadcast-testikanava (DVB-H)

Edit Scheduled Task

Exec
 HttpCall
 Program
 Interaction
 Dataset

Name: The Pacific

Start: 2010-08-31 Time: 16:00:00

End: 2010-08-31 Time: 17:15:00 duration: 01:15:00

Details | **Preview Data**

Title (eng): The Pacific

Description (eng): 3

Title (swe):

Description (swe): isLive=false, isReplay=true, isFreeToAir=true

Rating: [dropdown]

Genre: [dropdown]

Content Rights: [dropdown]

Figure 12 b)

Figure 12. Sofia Digital Backstage Service Manager management interface in a) channel settings and b) scheduling.

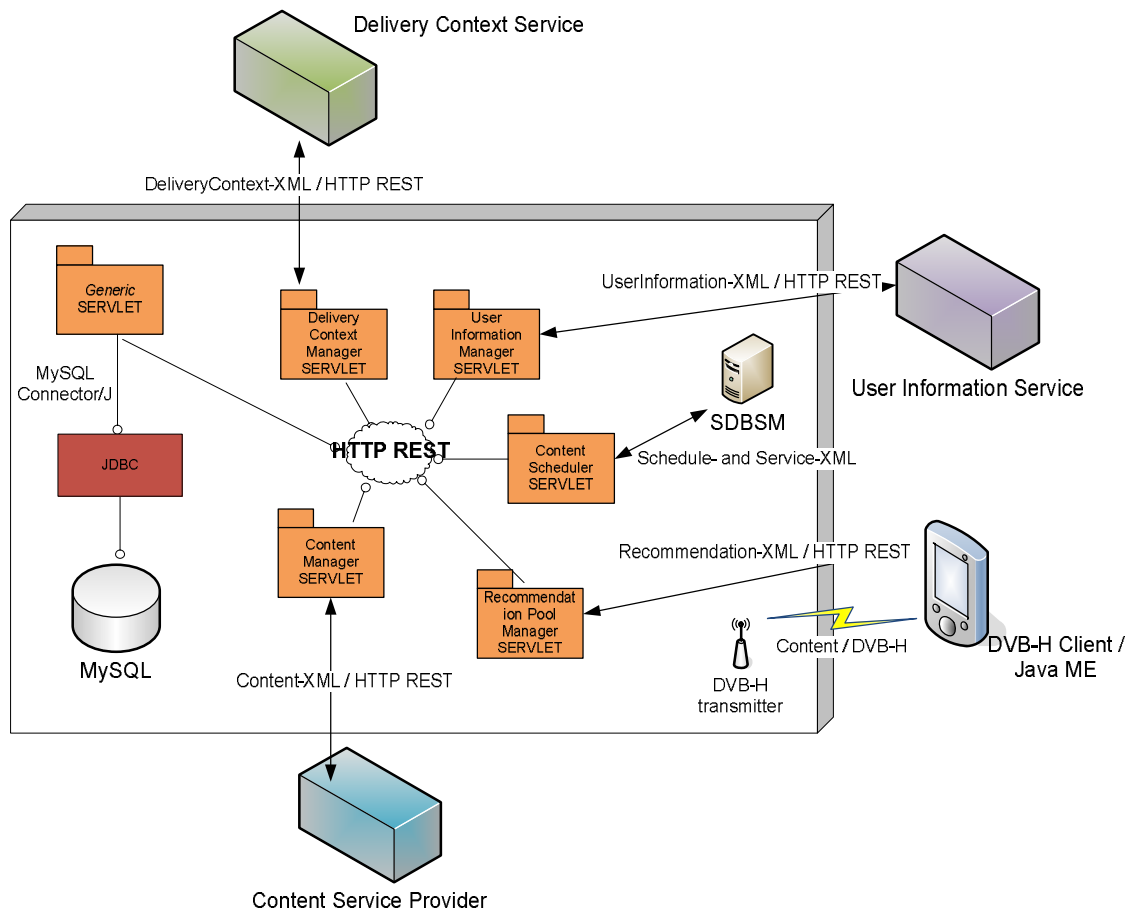


Figure 13. DVB-H Service overall implementation from a technological viewpoint. HTTP REST is utilized for information exchange between the entities of the framework. The database is accessed with MySQL Connector/J driver. DVB-H Client is implemented by using Java ME.

Next, the implementations of DVB-H Client and DVB-H Recommendation Service are explained separately with the help of Figures 14 and 15, which depict the different entities within the corresponding system and the information flow between them. Figure 14 conveys the basic principles of DVB-H Client. As user starts the application, a control UI is displayed with the possibility to hide or exit the application. Meanwhile, the software immediately starts the process of polling for new recommendations from DVB-H Recommendation Service. The new recommendations are received, parsed and a notification is generated with the parsed data. User is then alerted of the incoming recommendation, which is then displayed to the user. The dashed line components in Figure 14 represent the feedback system, which is to be implemented in the future for DVB-H Service.

Figure 15 considers DVB-H Recommendation Service in detail. Delivery Context Manager receives Context- and Profile-XMLs from Delivery Context Service. This information is transformed into DeliveryContext-XML, which is delivered to Inference Engine on request. Content Manager receives Content-XML from Content Service Provider, after which this information is sent to the database. Similarly, Content Scheduler receives scheduling information from Sofia Digital Backstage Service Manager. This

information is transformed into Schedule- and Service-XMLs, which are eventually sent to the database. Again, User Information Manager receives user information from User Information Service. UserInformation-XML is generated and sent to the database. Schedule Inspection Manager initializes the recommendation generation process by querying the database for relevant content. It then sends relevant schedule IDs and their recommendation types to Inference Engine. Inference Engine decides if a user is interested in this schedule item. If so, the recommendation data is forwarded to Recommendation Filter for further processing, finally ending up to Recommendation Pool Manager, where the actual Recommendation-XML is generated and sent to the Recommendation Pool, i.e., one of the tables in the database. DVB-H Client can now fetch the recommendation from Recommendation Pool Manager. For more detailed descriptions on the operating principles of Inference Engine, Recommendation Filter and Recommendation Pool Manager, refer to the figure in Appendix 3. The image contains pseudo-code to understand the recommendation generation process more technically.

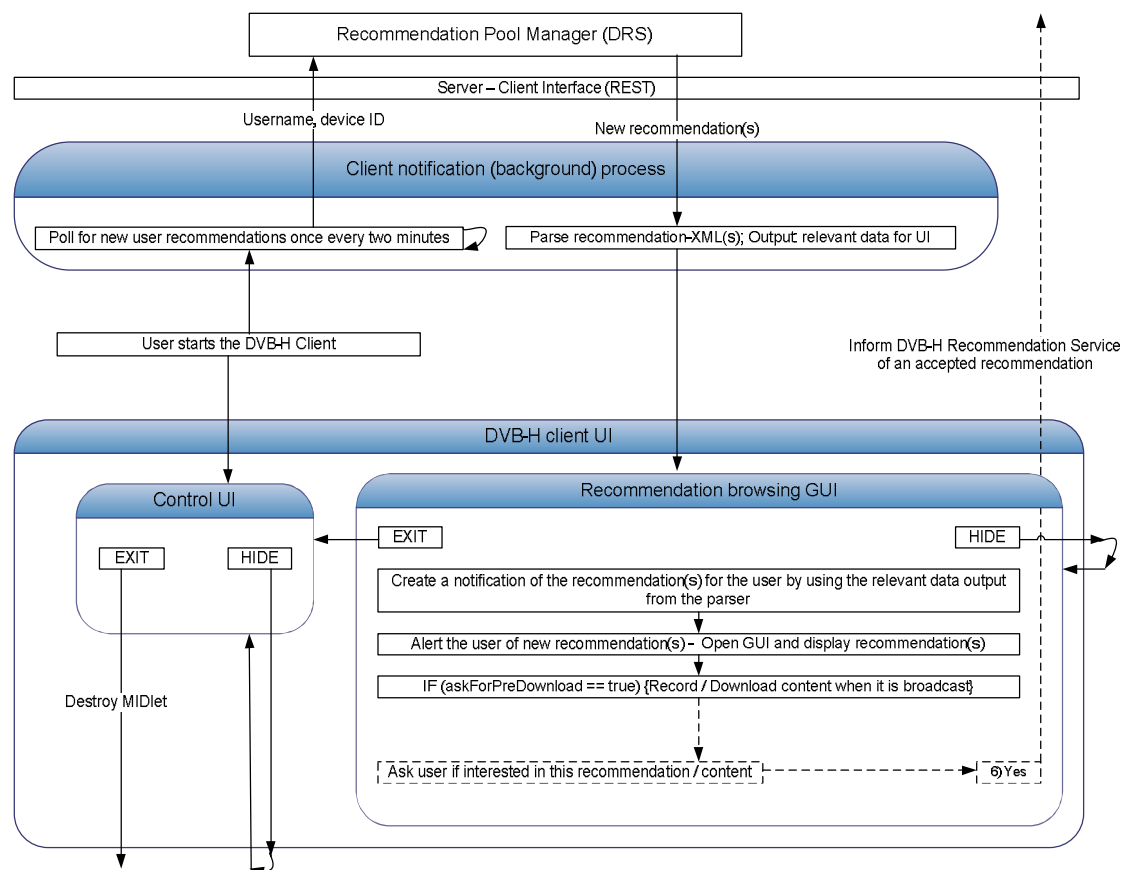


Figure 14. DVB-H Client implementation, once initialized, polls for new user recommendations from DVB-H Recommendation Service. Once new recommendations arrive, they are parsed and presented to the user by the user interface. Control UI is used only for hiding or exiting the client application.

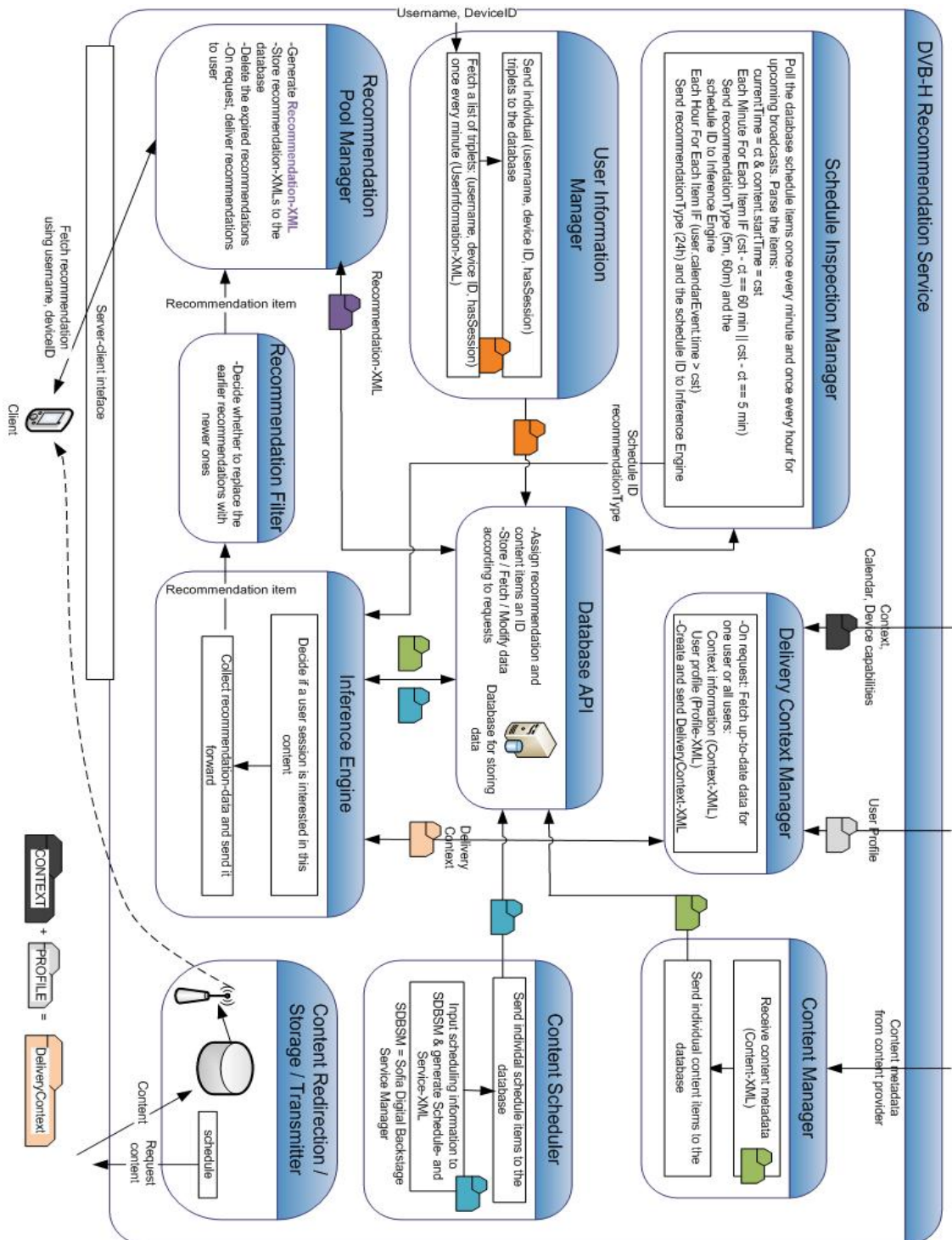


Figure 15. DVB-H Recommendation Service and its various functional entities. Content metadata is ingested into Content Manager. Content Scheduler receives schedule metadata. Delivery Context Manager is responsible for fetching user metadata. User Information Manager keeps track of the users of the DVB-H Service. Database stores all the relevant metadata. Schedule Inspection Manager is responsible for initiating the recommendation process, which is then completed by Inference Engine, Recommendation Filter and Recommendation Pool Manager.

5.2.1 Integration management interfaces

In the integrated implementation, the administrator of DVB-H Recommendation Service is responsible for creating and ingesting the content metadata, complying with ContentSchema, into DRS. Furthermore, the administrator is responsible for ingesting schedule metadata, complying with SofiaDigitalSchema, into DRS. The latter metadata is received from SDBSM. An interface for ingesting this metadata was designed and implemented for demo purposes, although it would be as applicable with an actual Content Service Provider. The ingestion operations can be performed from a simple DRS management interface, as shown in Figure 16. The metadata documents are uploaded locally into DRS, where they are further processed into the MySQL database. Some interesting and specific information regarding the details of scheduling can also be seen in Figure 16.

[Go to index page](#)

Content Metadata

All content-items' content IDs must be equal to '0'.

Please specify a content metadata file (must conform to ContentSchema.xsd):

[Selaa...](#)

[Upload content metadata](#)

Schedule Metadata

Scheduling is done manually in Sofia Digital Backstage Service Manager (DVB-H management interface), from which the schedule is exported as an XML-file (conforming to *SofiaDigitalSchema*). This XML-file contains references to the content IDs in question. These content IDs are input manually to Sofia Digital Backstage Service Manager calendar ("Description (eng)" field) by the person doing the scheduling. Likewise, the field "Title (eng)" shall contain the content title and the field "Description (swe)" shall contain three parameters in the following format:
 "isLive=*boolean*,isReplay=*boolean*,isFreeToAir=*boolean*."

Example - how to extract schedule/service meta-data from Backstage Service Manager:
http://130.188.225.244:8080/backstage_manager/feed/calendar.jsp?cid=195&ynd=20100828&days=7
 cid = serviceId (in the Sofia Digital schedule-XML the id-tag under channel-tag)

Please specify a schedule metadata file (must conform to SofiaDigitalSchema.xsd):

[Selaa...](#)

[Upload schedule metadata](#)

Figure 16. Metadata uploader in DVB-H Service. Content Service Provider uploads content metadata using the uploader, which mediates the information to Content Manager in DVB-H Recommendation Service. Schedule metadata is currently uploaded by the administrator of DVB-H Service for the use of Content Scheduler in DVB-H Recommendation Service. This is done after the administrator has decided when the content is to be broadcast and having input this information into Sofia Digital Backstage Service Manager.

When the user of DVB-H Service activates DVB-H Client, a simple control UI screen is displayed as shown in Figure 17. The UI can be hidden to the background so as not to be visible for the user; it operates automatically. When DC is running in the current implementation, the software queries DRS for new recommendations every two minutes. If a new recommendation(s) is received, the user will be informed of this by bringing DC GUI to the foreground in the form of an alert. This GUI is separate from the control UI, although they operate under the same MIDlet.

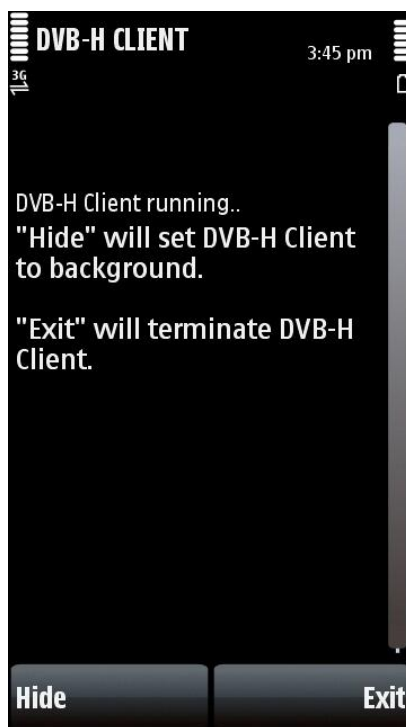


Figure 17. The control UI screen in DC, which is displayed when the application is running.

5.3 Reference implementation scenarios

To demonstrate the capabilities and the performance of Context-Aware DVB-H Service implementation in practice, several purposeful scenarios were designed and tested. The idea was to test the implementation's functionalities and inference rules, by which the recommendations are generated.

Some assumptions were made to make the practical implementation of Context-Aware DVB-H Service scenarios more feasible:

- DVB-H transmitters cover the whole area where the user is expected to operate while using Context-Aware DVB-H Service
- DVB-H Recommendation Service operator manages the content scheduling with Sofia Digital Backstage Service Manager

Additionally, in the finished reference implementation framework there was no actual Content Service Provider (Figure 8 in Section 4.1) per se. Instead, demo content metadata was elaborated complying with ContentSchema. In a commercial application, an actual Content Service Provider would provide a document similar to the demo content metadata. In a similar manner, User Information Service was assumed to exist (as part of the CoDe-project). For this implementation, a Java Servlet class, containing demo user data, was compiled for delivering the user information on request. Finally, in CoDe-project Delivery Context Service has been implemented. Due to the practical restrictions in the scope of this Thesis and lack of suitable service interfaces, the Delivery Context Service was not used as such in this implementation. However, all the metadata used in Delivery Context Service was used in this implementation to enable the compatibility of these services in the future. A Java Servlet class, containing demo user profiles and contexts, was compiled for delivering the delivery context information on request.

In this Section, the most essential implementation scenarios will be introduced. After each scenario introduction, the working mechanisms, which are utilized in the scenario in question, are described in detail. Especially Section 5.3.3 is important as it covers the most relevant details of the recommendation process.

5.3.1 Basic demo scenario

Below, the basic demo scenario is described. The operations occurring in the background are then elaborated.

Two DVB-H Service users, John and Mary, receive recommendations from DVB-H Recommendation Service for a program called “Helsingin seudun uutiset” starting in five minutes. The reason for receiving the recommendations is because their profile and context information match to this particular program: John expresses his interest in “Helsinki” and his main language is “English”, which is also the audio language of the program. John also wants to watch content that is short in terms of duration, meaning the program lasts for maximum of 10 minutes. It is presumed one would be interested in the program if one was in Helsinki. Mary’s context indicates she is now in Helsinki. Even though Mary’s mobile device resolution is inadequate for viewing this content (of which she is notified in the recommendation), the program can still be viewed, or at least listened to. In her interests, Mary actually expresses her interest towards this particular content, so it is automatically recommended to her.

What actually happens in DRS when these recommendations are generated? Firstly, it is relevant to know that the content metadata related to the program “Helsingin seudun uutiset” along with its scheduling and service metadata are now stored in the MySQL database. Secondly, John and Mary’s current delivery context information and their user information are available on request from Delivery Context Service.

The recommendation mechanism is initialized when Schedule Inspector Manager in DRS, through a database inspection process performed once every minute, detects the

particular program is going to be broadcast in five minutes. Schedule Inspector Manager then sends information about this occurrence to DRS Inference Engine. The two parameters being delivered are schedule ID and recommendation type; in this case the recommendation type is 5m, referring to the fact the program is going to be broadcast in five minutes.

Inference Engine receives the two parameters. First, it requests the delivery contexts of all the users of DVB-H Service from Delivery Context Manager. After receiving the request, Delivery Context Manager furthermore requests the user information of all the users of DVB-H Service from the external Delivery Context Service. Once the individual Context- and Profile-XMLs have been received by the Delivery Context Manager, it then processes this data to generate one single XML-document complying with DeliveryContextSchema. This document contains the delivery context information of all the users in sequential order. The document is then sent back to Inference Engine. Secondly, Inference Engine fetches content, schedule and service metadata from the MySQL database, based on the schedule ID: schedule ID is used to find the appropriate piece of schedule metadata. From this metadata, the service and content IDs can be located. By using these IDs, content and service metadata will be fetched. The recommendation type, the content, service and schedule metadata, and the delivery context XML-document are then sent forward to the class responsible for generating the actual recommendations.

The process of recommending content is performed for all users declared in the delivery context XML-document – in this case John and Mary. If there were multiple content recommendations to be done consecutively, each piece of content would go through the recommendation process individually, and always for all users. To start generating the recommendations, first a *technicalRecommendation* is either given or not given for the *recommendation item*. A recommendation item is a textual entity conveying relevant information regarding a recommendation throughout the process. This textual information may or may not be used by Recommendation Pool Manager at the end of this recommendation process to create the actual recommendation as an XML-document. A technicalRecommendation is required for the recommendation process to proceed any further for a single user. The user's client device is qualified technically against the content and broadcast data, e.g., MIME-types, available space or resolution are parameters that are inspected in the process. Some of these are critical and some are not: if the client device does not support a certain video format, it can not be displayed on the device, hence preventing the content from being recommended. On the other hand, available space can be too low for an application download; the content can be recommended but this time a warning related to this matter will be displayed for the user in DVB-H Client.

If a technicalRecommendation is given for the recommendation item, it is further processed. The next step is to compare relevant data from the current schedule item against the user's delivery context. The comparisons are made with specific rules, which have been given a certain amount of points. These points are used to decide whether this content item should be recommended to a user. The rules are conceived by the author of this Thesis. Currently, the comparisons between the XML-documents are done either by

plain matching of strings or then by inspecting if certain parameter(s) is(are) present, which would affect the recommendation points. The points can either be positive or negative. A threshold for the points has been defined: if the points reach the threshold or go over it, a recommendation of certain type can be given for the recommendation item. These recommendation types are described next for the relevant parts.

User's profile data is compared against the schedule item. If the comparison points are adequate, the recommendation item will be given a *profileRecommendation* for the user. As an example, if the user's interests are congruent with those announced in the content metadata, points for profileRecommendation are given. Enough of these points then lead to a profileRecommendation.

Next, the user's context data will be compared against the schedule item. If the comparison points are adequate and if the user has an active session, the recommendation item will be given a *contextRecommendation* for this user. Active session is required as there is no use in utilizing context information, which, most likely, is already obsolete. If the content requires certain contexts to be present, user contexts are examined to resolve whether the user has these contexts. The information regarding this will be stored for later use with a tag *hasRequiredContexts*. If the user has all the required contexts and has an active session in addition to reaching the threshold value, the content item will receive a contextRecommendation. If the content does not require any contexts to be present, *hasRequiredContexts* is always "true". Contexts might be required to be present, e.g., if the content is meant to be consumed in a certain place or at a certain hour.

The decision to forward the recommendation item is based on following. First, if there is a technicalRecommendation, the recommendation item may proceed. Next, if there is at least either a profileRecommendation or a contextRecommendation, the recommendation item is sent forward to Recommendation Filter. Actually, in the case of a recommendation item, whose type is 5m, the item will be forwarded to Recommendation Filter even if there is no profileRecommendation or contextRecommendation. The reason for this is that a recommendation related to this schedule item might have been generated earlier as well. In this case the forwarding of 5m-recommendation is needed to inform the end-user that the earlier recommendation is not relevant anymore.

When the 5m-recommendation item related to this scenario arrives to Recommendation Filter, the class inspects whether an earlier recommendation exists. If there is no earlier recommendation related to the current 5m-recommendation item, it is forwarded without any inspections to Recommendation Pool Manager. However, if an earlier recommendation does exist, the current recommendation item is either discarded or forwarded to RPM depending on several factors. One of the most essential factors is the question of whether the user has already fetched the earlier recommendation. The earlier recommendation may or may not then be deleted.

RPM uses the received recommendation item as the basis for generating the actual recommendation XML-document. This recommendation is stored into Recommendation Pool, where it is given a recommendation ID for reference. Finally, when either John or Mary runs DVB-H Client requesting new recommendations from DRS, RPM

fetches the respective recommendation from the database and sends it to DC in a response. The DC then processes this information to generate the recommendation, which is finally displayed to the user.

5.3.2 Museum demo scenario

John is on a vacation and just now visiting Alvar Aalto Museum. The museum area has an audio broadcast playing in a loop – a museum guide, which gives various details regarding the famous Finnish architect Alvar Aalto and his achievements, which can be seen in the museum collections. John has been admiring the artwork of Alvar Aalto for 20 minutes when he suddenly receives a notification of an audio guide, which is due to start in five minutes. Knowing this John now tunes his mobile device to receive the audio stream. John listens to the audio guide, complementing his visit and exploration of Alvar Aalto Museum.

The logic of creating the museum guide recommendation is similar to that of the basic demo scenario. This time the generation of recommendation emphasizes simply the location of the user, i.e., the museum. The user is expected to be interested in this piece of content, since he is in an exhibition. When John receives the recommendation, he is presented with what can be seen in Figure 18.



Figure 18. Museum guide recommendation, which is displayed to the user in DVB-H Client after receiving the recommendation from DVB-H Recommendation Service.

5.3.3 Car demo scenario

This scenario's time-span is extended, so it is divided into multiple parts. It is the most diverse of the scenarios presented.

60 minutes before the scheduled content broadcast

Mary is in Otaniemi, Espoo in a work meeting. Her active context at the moment thus is “do-not-disturb”. Meanwhile, DVB-H Recommendation Service notices there is a series called “The Pacific” coming into broadcast in exactly one hour and that Mary might be interested in this content. The service then also notices that Mary does not want to be bothered at the moment, so the recommendation is not delivered to Mary's mobile device.

The difference in the recommendation generation process in this scenario part compared to the earlier scenarios is that now the recommendation type is 60m. Had there been neither profileRecommendation nor contextRecommendation for this recommendation item, it would have been discarded already in Inference Engine. The process is otherwise similar to the previous scenarios. However, now a “do-not-disturb” context has been detected in Mary's delivery context. This means that the recommendation, which is in Recommendation Pool, will not be sent to Mary's DC before a recheck of Mary's contexts is done. Every time DC requests for new recommendations and the context “do-not-disturb” is active, RPM sends the recommendation for a recheck through Inference Engine. For this, a special type of recommendation called “instant” is used. Using this recommendation type, Inference Engine requests only Mary's delivery context from Delivery Context Manager instead of the delivery contexts of all users. Having received the most up-to-date context data of Mary's situation, Inference Engine then checks whether Mary has the required contexts and if Mary still has the “do-not-disturb” as an active context. This information is delivered to RPM. Based on this information, RPM makes the decision on whether to send the recommendation to Mary. In this part of the scenario the recommendation is not sent.

30 minutes before the scheduled content broadcast

Mary is on her lunch break but spending it on some personal errands: she is on her way to pick up her car that was under maintenance in Helsinki. Mary is on a bus in Helsinki when DVB-H Recommendation Service notices that the “do-not-disturb” context is no longer valid. The recommendation regarding the series “The Pacific” is sent to Mary. She is pleased to know the program starts in 30 minutes, although she is in a bit of a rush. Mary then arrives to the destination and puts his mobile device to her pocket.

The information Mary sees when she has received the recommendation is illustrated in Figure 19. In Figures 19 a) and 19 b) some basic information along with multicast IP/port are displayed. By choosing to see “Additional Information” Mary sees the contents of Figures 19 c) and 19 d). This is supplementary information for the more interested users of DVB-H Service.



Figure 19 a)

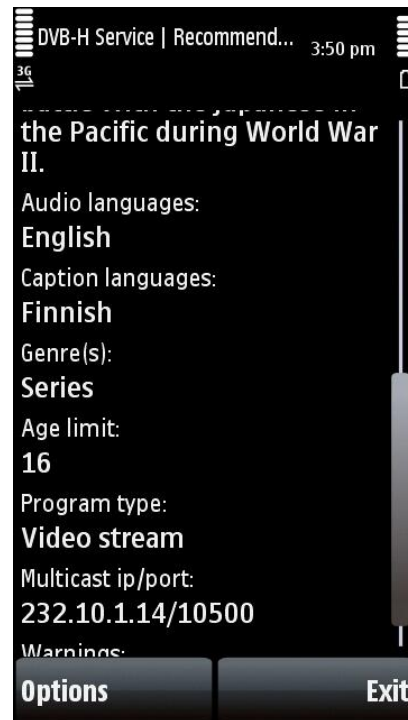


Figure 19 b)

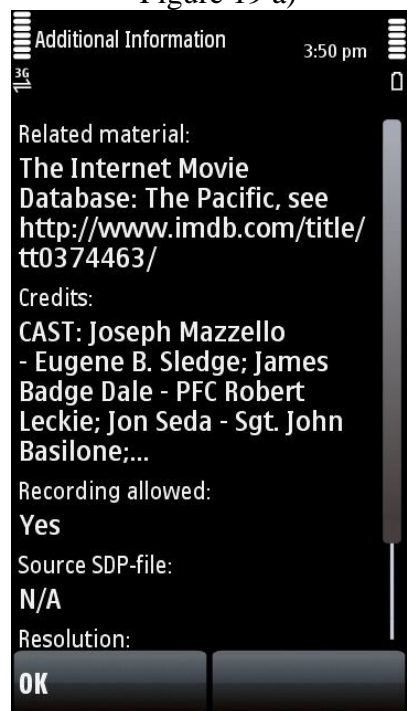


Figure 19 c)

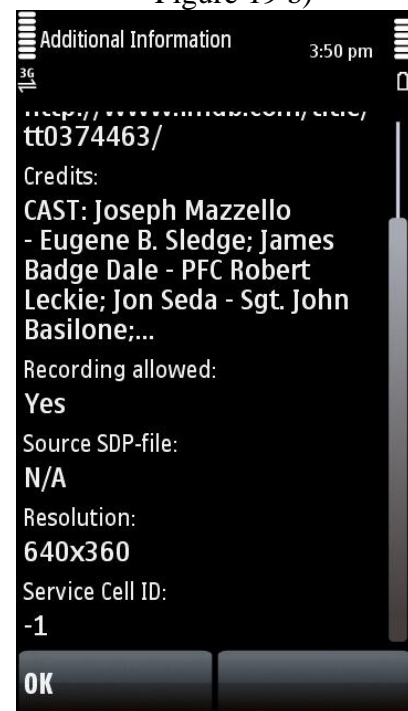


Figure 19 d)

Figure 19. The recommendation related to TV-series "The Pacific" 30 minutes before broadcast. Figures a) and b) contain basic information, whereas c) and d) offer supplementary information.

5 minutes before the scheduled content broadcast

Five minutes before the scheduled broadcast of the series “The Pacific”, Mary is rushing back to work in Espoo with her newly picked up car. DVB-H Recommendation Service notices Mary is in a car so she is effectively unable to view the series safely. Instead, Mary is notified of this hazard and asked whether she would like the series to be recorded automatically. Mary accepts this offer and continues driving back to her work safely.

In this scenario, a special rule is applied. If a user has the context “car” and if the content being recommended contains video, the user will be notified of this potential hazard. The user could also be a passenger in the car but highly unlikely: the inference done in Delivery Context Service assumes that when a user is in certain motion and additionally has a headset attached to the mobile device, the user is driving a car. Mary now sees the warning related to the impending fact. She then chooses the option, which enables the (not implemented) feature responsible for automatic recording of the video stream. These events are illustrated in Figures 20 a) and b).

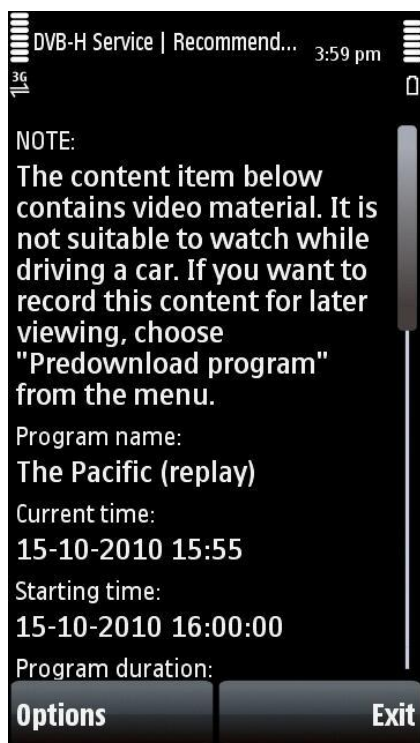


Figure 20 a)

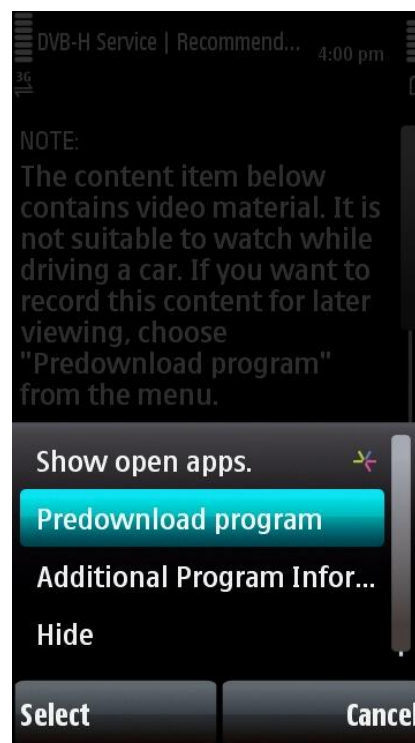


Figure 20 b)

Figure 20. The recommendation related to TV-series “The Pacific” 5 minutes before broadcast. Mary is in a car so the recommendation proposes alternative ways of viewing the series, i.e., recording the program. In a) the notification related to the hazard and in b) “Predownload program” effectively means the recording of the program.

5.3.4 Predownload demo scenario

John is at home in the evening of 31.8.2010. John's mobile device calendar indicates an event, which is supposed to take place the next day (1.9.2010) at 13:00 in Otaniemi (a visit to a museum). John now starts the DVB-H Client.

DVB-H Recommendation Service has noticed previously, at 13:00 on 31.8.2010, that John has a relevant calendar event on the next day at 13:00. DVB-H Recommendation Service checked all the upcoming content items from 14:00 on 30.8.2010 until 13:00 on 1.9.2010. A content item called "Alvar Aalto Info Package" is found to be broadcast in a DVB-H file carousel in the time interval of 04:00 to 05:00 on 1.9.2010, i.e., in the middle of the night. This content item's keywords include Otaniemi and Alvar Aalto, which are the words found from John's calendar event and profile information, respectively. Thus, DVB-H Recommendation Service has a ready recommendation waiting for John as he starts the client.

When John starts the DVB-H Client in the evening of 31.8, he receives a notification of the content "Alvar Aalto Info Package" and is asked whether he would like the content to be downloaded automatically for him to be able to consume it on the actual calendar event's time. John accepts this offer. The notification also informs John that the available space on his mobile device is running low so he should delete something before the download can be performed successfully.

The recommendation type in this scenario is "24h". From the standpoint of the recommendation process, a 24h-recommendation is considered separate from the other recommendation types. First of all, the process of checking content for this type is performed only once in an hour. This process begins with checking the database for schedule items that will be broadcast in the upcoming 24 hours. When the items are found they are sent to Inference Engine, as usual. This time Inference Engine only ensures the recommendation item has a `technicalRecommendation` and a `calendarRecommendation`. A `calendarRecommendation` is given if the user has calendar events with interests towards any content coming into broadcast before the event.

The recommendation John receives in the evening of 31.8.2010 is displayed in Figures 21 a), 21 b) and 21 c). Note from Figure 21 b) how DRS has noticed John has insufficient space for downloading the recommended content item. John is able to tell DC to predownload the program in the night time, so he is able to consume it on the next day's visit to the museum.

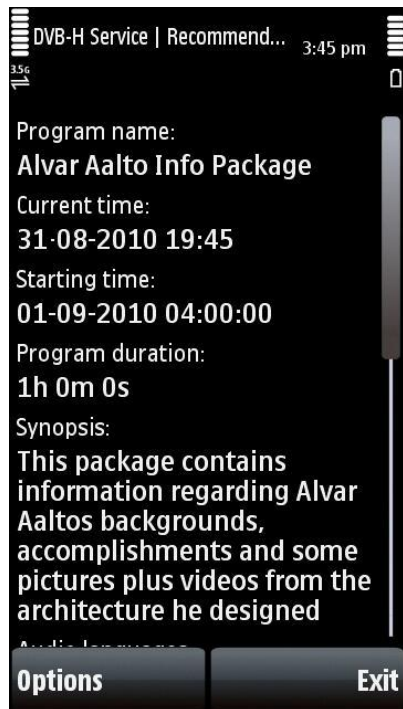


Figure 21 a)

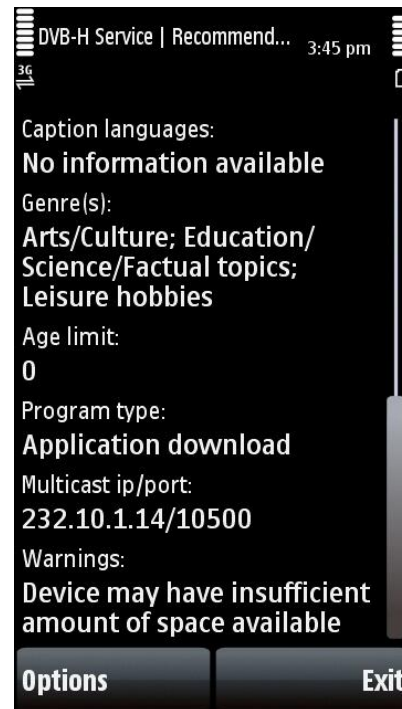


Figure 21 b)

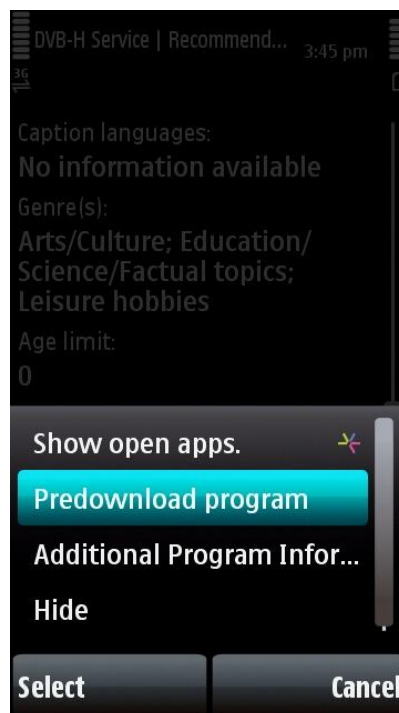


Figure 21 c)

Figure 21. The recommendation related to the broadcasting of the application "Alvar Aalto Info Package". The content is something the user might be interested in based on her upcoming calendar events. In a) and b) the basic information related to the recommendation with additional warning and c) the option for predownload is presented.

5.4 Verification of software framework requirements

The presented scenarios all demonstrate DVB-H Service is able to offer recommendations to its users in a way that considers the context and the preferences of the users. This was one of the main requirements originally set for the developing of Context-Aware DVB-H Service Software Framework. For this requirement, the Delivery Context Service, also present in CoDe framework, was utilized.

An actual content provider component was not implemented other than as a simple class offering demo content metadata. However, for DVB-H Service the only requirement was that it should receive precise and descriptive content metadata. This requirement is satisfied by using ContentSchema, which was elaborated partly on the basis of DVB-H ESG Schemas depicting relevant keywords, and also by including keywords, which describe the context where the content is intended to be used. A similar approach was used with User Information Service, although it is significantly simpler. For scheduling, Sofia Digital Backstage Service Manager was utilized. It provided adequate features for versatile content scheduling and for exporting this information for the use of DVB-H Service core services.

The scenarios presented in this Chapter are more complex than the scenario presented in Section 4.1 for deriving the requirements. Additionally, the presented scenarios include implicitly all the events of the requirement scenario. Thus, verification of the more complex scenarios is sufficient to verify the requirement scenario.

The presented scenarios display and depict DVB-H Client application. It was kept as simple and comprehensible as possible. Information overload was avoided by showing the recommendations only once to the user, and the amount of information per recommendation was kept at the pure essentials. Feedback was sufficient by using various dialog boxes to keep the user up-to-date of what is going on. Privacy and the adjustability of level of automation were not considered, since they were left out from the requirements. Next, the presented scenarios are discussed in more detail to grasp how they meet the requirements set for the developed software framework.

The first of the four presented scenarios uses relatively simple logic for the deduction process. The main factor considered here is location context and various preferences, which in some way indicate that the users are interested in the recommended content. The second scenario is more location-specific, as DVB-H Service here clearly has been designed for the museum guests. The location itself, and an assumption that all guests are by default interested in additional information regarding the exhibition, is well enough information to compile the recommendation here. The third scenario considers “the small details” in DVB-H Service. The utilization of the “do-not-disturb” context means DVB-H Service functions in a situation-aware way, since it understands the user is not to be disturbed. Another point, where situation-awareness can be seen, is when DVB-H Client proposes the user that the series could be recorded, since she is most likely driving a car. This feature, like many others, was implemented only after the scenario had been tested once already. The development of these specific features appeared

to result from observing what requirements situations impose for DVB-H Service to function in context-aware and intelligible way, even more than by plain requirement observation. The last presented scenario demonstrated DVB-H Service's capability to utilize calendar information for inference. The main reasoning here was that some content might be relevant for user only at some defined point of time in the future, but not necessarily at the moment. When a calendar event indicates the location and keywords for this event, DVB-H Service is able to make recommendations based on that information. It is done early enough (maximum of 24 hours prior) to maximize the amount of content, which can be compared against the calendar event metadata. On the other hand, this time is close enough to the actual event for the user to understand the recommendation is relevant to her.

As the scenarios show, personalization is taken into account by considering information related to the users, such as their age, gender, language, interests, disinterests, etc: the relevance of recommendations is inferred, and the recommendations are compiled by utilizing this information. Similarly, context information, such as location, time, activity, client device capabilities or calendar information is taken into use when deciding whether the broadcast content is relevant for the user. Both user profile and context information are refined in the inference logic of DVB-H Service to understand the situation of a user better. This is made possible by well-designed metadata information representations. DVB-H Service Inference Engine is able to utilize these representations for the benefit of inferring whether user is interested in some content.

It can be questioned, whether the decisions to recommend content are valid. This is because the final decisions are based on rules, which are not conceived by any standard guidelines. However, considering what the users of the depicted scenarios would feel about the recommendations they received, and by using basic reasoning, it is intuitively clear that they were, situation-wise, interested in receiving these recommendations. As the objective of this Thesis was to create a functioning recommendation service for DVB-H-mediated content, the requirements are thus fulfilled for this part.

5.5 Summary of the implementation

The technologies chosen for Context-Aware DVB-H Service Software Framework implementation are simple and well-known. They contribute in making the service very light-weight in terms of computational complexity, especially in client-side. Additionally, the technologies render the use and the possible modifying of the service straightforward. They offer enough functionality for the purposes and needs of the developed software framework, since it was originally meant to be extremely comprehensible and easy to grasp conceptually. The client implementation is plain and usable, and it does not interrupt the user any more than is absolutely necessary.

As the presented scenarios prove, DVB-H Service is capable of serving dynamically relevant recommendations for the users in multitude of situations. The scenarios are derived from something that could be a real-life situation. This, and the successful testing of the scenarios, indicates that the developed service would actually provide essential

recommendations for the users of DVB-H Service. The restrictions in the implementation are related to the ways of controlling the behavior of the software from the viewpoint of the service user. This is related to insufficient technologies available: e.g., pre-downloading a program is currently not possible as there are no implemented Java ME specifications for actualizing this in DVB-H. Due to restrictions in time and the limited scope of the Thesis, the reference implementation is relatively simple, and level of automation is non-adjustable. In addition, utilizing ontologies might have been beneficial had there been more time and expertise to implement this type of information representation.

6 DISCUSSION OF THE DEVELOPED SOFTWARE FRAMEWORK AND REFERENCE IMPLEMENTATION

The developed software framework was designed to be simple and easy to understand conceptually, while the implementation was actualized with relatively robust and easy-to-use technologies. The framework is meant to be used in tandem with DVB-H, but it could easily be perceived to utilize some other content medium as well. Considering the scope of this Thesis, both the framework and the actual implementation have well achieved the goals set for this Thesis: to develop and actualize a functioning small-scale context-aware DVB-H service.

In the following, the developed software framework and the reference implementation both are evaluated by reflecting their properties against the requirements, which were established in Section 4.1. Also, a wider perspective is considered by taking into account all the guidelines presented in Chapter 3. After this, the restraints related to these achievements are pondered, and finally some future views are elaborated to further improve the accomplishments of this Thesis.

6.1 Evaluation

The software framework presented in this Thesis is a small-scale recommendation delivery system with context-aware functionality and the possibility for DVB-H content delivery. The process of developing the framework, while heuristic by nature, provided valuable insight and a strong basis for future development of context-aware DVB-H recommendation techniques. The framework architecture resembles somewhat to those of FLAME2008 and MyTV 2.0, described in Sections 2.6.2 and 2.6.3. This shows the architecture used here is well-approved. At its core, Context-Aware DVB-H Service is a plain server-client software framework, which is complemented by several external services for allowing context-awareness and personalization to be used in the content mediated by DVB-H technology.

Context-Aware DVB-H Service reference implementation was actualized to demonstrate that for the users of DVB-H Service relevant DVB-H content can be recommended by utilizing context-awareness and personalization. The practical functioning of the implementation was confirmed by preparing several test scenarios, which were put to test. The tests verified the implementation is capable of generating and offering relevant content recommendations for the users based on their interests and current context. The performance of the implementation was satisfactory considering the scope of the Thesis. There is novelty value in the implementation, since there exists only few similar implementations (c.f. earlier work presented in Section 2.6), and DVB-H needs to develop its ability to offer more relevant content for both the providers and the users of such service in order to gain added value. The implementation in many ways is inadequate, but provides a good starting point for improving and adding the functionalities that are needed to elaborate DVB-H into a more attractive content delivery medium.

By using the definitions from Section 3.1, Context-Aware DVB-H Service can be classified as 1) one-to-many, 2) information, 3) user-received and 4) media industry mobile service. Furthermore, DVB-H Service supports the presentation of information and services to user but does not give the option for tagging of context to information for later retrieval. The latter is a relevant property and should be included in the future versions of DVB-H Service. Automatic execution of the service regarding the generation of recommendations is featured in the framework, but currently starting the receiving of the actual broadcast in the client software is absent.

When comparing FLAME2008 (cf. Section 2.6.2) to Context-Aware DVB-H Service, one may conclude they have a lot of similarities. There is some prestige in this as FLAME2008 was used in Beijing Olympic Games in 2008. As FLAME2008, DVB-H Service utilizes the concept of individual push of meaningful offers for information and services to the mobile frontend. Furthermore, in FLAME2008 the recommendation mechanism is based on offers and demands, which are matched against each other. The similar principles are also used in DVB-H Service: content and schedule metadata can be considered as offers, while user context and profile metadata can be thought of as demands. Considering Figure 4 depicting the general system architecture of FLAME2008, and comparing it to Figure 8 depicting DVB-H Service framework, the similarities become obvious. Information logistics in Figure 4 correspond to Inference framework present in DVB-H Service. User Context Service in DVB-H Service remains as separate service in FLAME2008 as well. FLAME2008 uses ontologies as the information representation model whereas XML Schemas are used in DVB-H Service. Service roaming for wide geographic coverage could as well be utilized in DVB-H Service for better user experience.

In MyTV 2.0 (cf. Section 2.6.3) a dynamic TVOntology was utilized. Here both content providers and users could evolve the ontology, whereas in DVB-H Service vocabulary is more static and the users are not able to participate in the development of the vocabulary. An assumption is made in DVB-H Service that all the players in the framework are fully aware of the used vocabulary, so in a way it is not an issue. However, very few things are in reality static, so modifications to information representation should be more flexible in DVB-H Service. The tags, which describe content information in MyTV 2.0 and correspond to the recommendations in DVB-H Service, are delivered via DVB-H. In DVB-H Service this data is delivered via ordinary 3G network. This is beneficial, as it allows more interactivity between the client and the service. Nevertheless, in MyTV 2.0 3G is also utilized: the users can send their own tags to the service provider by using 3G. This kind of interaction is absent in DVB-H Service. What is furthermore missing in DVB-H Service is the cooperative architecture of MyTV 2.0, i.e., group filtering via the use of social groups. An essential point, which is present both in DVB-H Service and MyTV 2.0, is the generation of recommendations: for the most essential parts the recommendations are done server-side, which means receiver resources are spared for more important tasks. To be more specific, in MyTV 2.0 there is still some individual filtering done in client-side. Substantially, DVB-H Service is all about making individual filtering server-side, since there are no groups or group filtering applied.

Notification Service for DVB-H Mobile Broadcast was presented in Section 2.6.4. The foundational principle behind it is practically the same as in DVB-H Service – to offer timely notifications related to content in DVB-H network. In the Notification Service for DVB-H Mobile Broadcast, DVB-H ESG was extended with additional information blocks. The notifications were represented by these blocks, so the Notification Service operates only in DVB-H domain as opposed to DVB-H Service. What is different between Notification Service and DVB-H Service is that in the former the user is able to subscribe to specific service-related notifications, while in the latter there are no such categories for different services. In DVB-H Service everything that is perceived as potentially interesting to the user is uniformly offered to the user. This is something DVB-H Service could be improved upon: to offer more specific recommendations via the use of service-related notifications.

From the perspective of ordinary mobile TV in terms of user acceptance, Context-Aware DVB-H Service coincides with several requirements mentioned in Section 3.2. The developed framework addresses individual needs of users in many typical usage spheres. This is done by offering few relevant services to users instead of implementing collections of useful features. The client application of DVB-H Service is easy to use and offers highly topical content recommendations. Using DVB-H as the means for delivering data, DVB-H Service is free for its users, which further increases the attractiveness for using this service.

Context-Aware DVB-H Service addresses many of the requirements mentioned in Section 3.3, which describes the essential issues encountered in designing similar mobile services and applications. Proactivity is one of the key benefits of DVB-H Service. Being highly proactive, DVB-H Service performs actions for the user while maintaining sufficient level of feedback: the client application shows necessary information for the user. This way the user always knows what is going on. On the other hand, this level of proactivity results in little user control with regard to adjusting the amount of notifications and dialogues. Privacy of users is not considered in DVB-H Service, which is a big deficit and should be dealt with in further research. On the other hand, information overload is not a problem in DVB-H Service. Users are notified when it is necessary and never more than once for the same piece of content to be broadcast in the near future.

DVB-H Service has many usability risks. One of the most prominent is the uncertainty in context recognition, or more accurately, the inference process leading to high-level context. High-level context is used to infer the situation of the user, which is a difficult task in any application. Since the rules, by which DVB-H Recommendation Service decides whether to recommend content to users, are ultimately based on heuristics, the validity of recommendations is debatable. Other issue is the level of feedback: DVB-H Service might be regarded as more trustworthy, if it were to offer, for example, some information related to the background processes of the underlying system. Definitely the lack of standardization is a big issue as well in DVB-H Service, especially when considering information representations. As this may lead to poor interoperability with other services, it is something that should be improved upon. Finally, DVB-H Client is unable to adjust itself in real-time to react to continuous changes in context. This

is a desirable feature in any modern mobile application. Then again, DVB-H Client UI is designed to be very simple, concise and easy to navigate, which appeals to modern users, who need to access information effortlessly, while at the same time using a device with relatively small display and buttons.

In Context-Aware DVB-H Service the low-level context sensing, and to some extent the high-level context inference as well, is inherently hidden from DVB-H Recommendation Service: User Context Service carries out both low-level context sensing and partially higher-level context inference. This context information is delivered to DVB-H Recommendation Service, which performs even further higher-level context inferences, namely making DVB-H Service a situation-aware service. In Section 3.4 this principle of hiding context sensing from the actual application is considered as a method for designing a successful context-aware mobile service.

Context-Aware DVB-H Service can be evaluated in relation to the design guidelines proposed by Dey and Häkkinen [11] in Section 3.4. As mentioned, proactivity of DVB-H Service is relatively high, even though the situation inference, or context-recognition, is more or less uncertain. Less proactivity might be worth reconsidering to alleviate this controversy. User control in the client application is implemented and ensured by various confirmation dialogues and possibilities for controlling the visibility and functioning of the application. Unnecessary interruptions are minimized in the client application so as to increase user performance and satisfaction of the whole system. Only relevant information is displayed to the user, when a recommendation is presented. System status is simple to manage: either the client is running or not, i.e., user is able to choose whether she wants to receive recommendations or not. However, there are no means for the user to abort the operations performed in DVB-H Recommendation Service. This means that regardless of user's will, recommendations are generated in server-side – something to be improved upon. Personalization can be considered from multiple aspects. However, individual needs in DVB-H Service are met inherently, as user preferences are always taken into account while generating recommendations. This assures personalization is always considered appropriately. The securing of users' privacy has not been considered in designing the software framework. Thus, it is a drawback waiting to be corrected. Social context in DVB-H Service is foreseen in some cases, i.e., if certain device or user behavior is inappropriate. For example, if the user has the context "do-not-disturb", the recommendation will not be sent to the client application. In addition, if the user is driving a car and about to view content containing video, she is notified of the hazard and given the choice to automatically record the program.

6.2 Considerations and future improvements

There are many challenges related to designing a context-aware service, which utilizes DVB-H as the exclusive content delivery medium. Inherently, DVB-H is a broadcast medium, which in practice means a one-to-many mobile service. DVB-H network, composed of multiple DVB-H transmitters with extensive geographical coverage, delivers some amount of services to an area. The services can be video streams, audio streams or data streams – each of which has its unique multicast address (IP/port).

However, the amount of services is not sufficient to cater for each individual user's personal preferences and context in that area. An interesting question is then raised: how to make a DVB-H-based service completely context-aware? If there was only one user in the whole DVB-H network, it would certainly be effortless to offer perfectly relevant content via DVB-H to the user. In reality, DVB-H would become an unprofitable means of delivering data if such scenario was true. When the amount of users in DVB-H network increases significantly compared to the amount of services offered, it becomes more and more difficult to serve content, with which all the users would be satisfied. The issue here is then how to offer relevant content *on average*. The phrase "everyone gets something – nobody gets everything" seems to be the only option. Unfortunately, this is not acceptable in the future network environment, as services need to be completely personalized to add value to a service. However, DVB-H is able to reach out to masses of people with practically no cost, which makes it a very attractive opportunity for cost-efficient means of content delivery. Consequently, in order to harness the potential of context-aware DVB-H, persistent research needs to be conducted to solve the problems at hand. Context-Aware DVB-H Service approaches this problem and presents a novel way of enhancing the current way of utilizing DVB-H in context-aware manner.

The used vocabulary and the way of representing information in Context-Aware DVB-H Service Software Framework utilizes currently markup scheme model as XML Schemas. The downside of using XML Schemas is their inability to capture complex conceptual relationships and constraints. In many cases, such as in DVB-H Service, XML Schemas were elaborated on the intuition of the author. Even though vocabulary from DVB-H ESG documents was utilized for the use of widely accepted terminology, the final form of the used XML Schemas is non-standardized, which makes their extensive usage in other applications relatively inflexible. Ontologies, while laborious, would provide DVB-H Service the ability to describe more complex and detailed relationships between the concepts used in representing profile and context data. To some extent, vocabulary unification would also be improved by using ontologies in information representation. In many cases ontologies are public and their development is a constant and never-ending standardization process, which consequently keeps the ontologies up-to-date even on a global scale. On the other hand, using ontologies can soon become too heavy for the mobile devices. For alleviating this, perhaps some kind of trade-off could be made between the two information representation models: utilizing ontologies to the fullest extent in the server-side, while the mobile clients could operate using XML Schemas, specifically tailored to reflect the complex relations of the server-side logic while still maintaining a lighter way of representing information.

Context-Aware DVB-H Service would benefit greatly from the utilization of context history; currently it is not applied. When recommending content, context history can be used to make inferences of the kinds of content the user is interested in: history of the user or history of users similar to the user can be considered as a good indicator of the user's future preferences. Although context history is not utilized in DVB-H Service per se, content-based filtering is used to some extent. The recommended items are selected based on the correlation between the content of the items and user preferences. However, earlier recommendations are not taken into account in the present software frame-

work. This could be done to improve the prediction accuracy in DVB-H Service. A mechanism for expressing explicit interest in particular content by the user would be required: this way DRS would understand what content is really interesting to the user. The user should be able to tag, or even rate, the kinds of content she is interested in to enhance the accuracy, with which the recommendations are made. DVB-H Service could implement a Facebook-style “Like”-button or a scale from one to five for every recommendation. When the user would press “Like” or choose a rating, information of this would be sent to DRS database of historical preferences. Going further with the idea, users might even be allowed to participate in the development of the used context vocabulary itself. On the other hand it would initially increase the complexity of inference significantly, but it would also offer the users of the service more unique service experience and increase the motivation for using it. In addition, after some time the partially user-created vocabulary would converge to sufficiently simple level of complexity as new words would become more and more exceptional.

Furthermore, DVB-H Service prediction accuracy would be substantially improved by utilizing collaborative filtering. This would require considerable modifications to the existing software framework, since the process of recommending content would also have to account for the preferences and contents of interest of similar users in DVB-H Service. A database for collecting and maintaining this information of all users would be essential. The basis for making these inferences could be in, e.g., dynamic interest groups, or social groups, which automatically would modify their organization to correspond to the changing set of data received from users and their indications of interests. The problem here is how to define what characteristics make up a group, i.e., in which situations can certain group be used for making a specific recommendation. Solution to this might be to use “interest group vectors”, which would self-organize themselves to resemble the average user. Then, in inference process, the user’s interests would be correlated to all of the interest group vectors in DVB-H Recommendation Service – the closest match would be considered as the most essential and probable, according to which the recommendation(s) would be made for the user. For the sake of minimizing computational complexity in the client device, this group filtering should be performed in server-side. Furthermore, individual filtering for further refinement of recommendation accuracy, at least to some extent, could be performed in client-side. Another option would be to utilize similar principles to those in C-CAST (cf. Section 2.6.1).

Extensions to DVB-H Service might include a feature in DVB-H Client, which would enable the user to initiate the recommendation process herself by sending keyword(s) to DVB-H Recommendation Service. DVB-H Recommendation Service would seek keywords among the content metadata, which would match the keywords the user supplied. The content should be such that it is going to be broadcast soon. The user would then receive a recommendation regarding this content. Although this would partially defeat the purpose of DVB-H Service being fully automatic, it would be a handy addition in many situations for most of the users. Another way of tailoring DVB-H Service would be to offer the user a possibility to subscribe to specific service-related notifications, as was done in Notification Service for DVB-H Mobile Broadcast (cf. Section 2.6.4). A major drawback currently in DVB-H Client is the fact that currently the user will only be notified of the incoming content, and the user will then have to seek inde-

pendently the recommended content by using other applications. This is one improvement required to make DVB-H Client more user-friendly. Finally, the absent privacy considerations in DVB-H Service need to be addressed in order for it to be perceived as a reliable mobile service.

Customization of the client is also necessary: for example, the feature for fetching recommendations automatically should be adjustable for the users inclined to controlling the software. Other interesting extensions to DVB-H Client include the checking of the DVB-H base station ID in frontend to improve quality of service, or implementing the predownload-feature, which, at the moment of the writing, was not possible with current standard Java ME technologies. On the whole, DVB-H Client should respond in a more dynamic way to changing situations – currently it is relatively static application. Dynamical operation would improve the user experience of the whole service. Depending on the preferences of user, more feedback from server-side might perhaps be a feasible scenario: users appreciate varying levels of proactivity. Currently, there is no function for the users of DVB-H Service to abort the operations related to DVB-H Recommendation Service recommendations. It is something the users would definitely appreciate.

Presently, Context-Aware DVB-H Service utilizes user profiles and contexts as well as content and scheduling metadata as the basis for making recommendations. In order to fully understand the situation, in which the user is, higher-level inference has to be rendered by using the mentioned data. These inferences are made according to the rules defined by the author of this Thesis. This is problematic, as there are no standardized or uniform practices for elaborating the rules. The rules would need to be confirmed either by further scientific research or by massive amounts of test data, whose statistics would reveal if the rules actually function the way they are expected in practical situations. In analytical sense, the rules are inherently troublesome as there is theoretically infinite amount of logical combinations of the rules and assumptions, by which a rule can be said to conduct its function. More certainty in the context recognition of DVB-H Service is required.

Context-Aware DVB-H Service can be used by a DVB-H service operator as an external piece of software, which enables the use of context-awareness in the whole DVB-H content consumption paradigm. It is meritorious to have the developed software framework as a semi-independent entity instead of being too profoundly integrated with DVB-H, since potentially it can also be used for other types of content delivery mediums for recommending content, although this would require modifications to the existing framework.

To establish a truly context-aware DVB-H service, it might be necessary to find an optimum solution for balancing the costs of creating the DVB-H network and the profit received therein. For example, small local transmitters might be utilized to transmit content covering a smaller area and thus a smaller amount of people with less variance in their preferences. Obviously, the location of the transmitter would be one context source, but more would be needed. Dynamic group-aware DVB-H service might serve this purpose. The context inference machinery would constantly modify the group pref-

erences according to the preferences of all the users within the coverage area, thus always transmitting the optimal content in the specific situation.

6.3 Summary of the discussion

After comparing the developed software framework and its reference implementation with the requirements of modern context-aware mobile services and applications, it is clear there is a lot to be improved upon. While several of the requirements were certainly met, few fundamental requirements were absent. Considering the objectives of the Thesis, i.e., to elaborate and cultivate the concept of DVB-H service that is context-aware, the software framework achieved here can still be considered as a success. In layman's terms, what it really is that was developed is a DVB-H service, which is both affordable for all parties involved and also now one step closer to reaching context-awareness with its full potential. Despite the hardships, the author of this Thesis feels that something authentic and novel was really created here.

7 CONCLUSIONS

The further we look into the future, the more the concept of context-awareness will be seen as the key property of any mobile service. Users of a mobile service demand content that is suitable for them in any situation. Therefore, rigorous research is needed to overcome the future challenges related to context-aware services and applications. Likewise, the ever-growing amount of the users of these services forces the service providers to search for affordable ways of delivering the content, of which these users are genuinely interested in. DVB-H technology can be considered as a very good alternative for delivering content to mobile devices in a scalable and inexpensive way. The facts mentioned above necessitate the need for a DVB-H-based service, which is able to utilize context-awareness and personalization for the benefit of both the users of this service and the service provider itself. The users get relevant content and see it as something well worth using, while the service providers get decent revenues from satisfied customers and from lowered network expenses.

The research problem of this Thesis was to determine how to utilize context-awareness and personalization to recommend relevant DVB-H-mediated content to users. To answer this question, the relevant technologies and concepts, such as DVB-H, context-awareness and personalization, were first described. This was followed by taking a look at some earlier work. This earlier work provided some novel ideas of how a context-aware DVB-H-based software framework could be attained. Essentially, the challenges and guidelines related to designing context-aware mobile services and applications were elaborated, which were finally used to establish the requirements for the software framework design. These requirements were used to verify and evaluate the software framework and the reference implementation achieved in this Thesis. The motivation behind this work was based on the need for the CoDe-project to create a context-aware mobile service utilizing DVB-H for media delivery. One of the main requirements in the creation process was to use the Delivery Context Service component, from CoDe framework, in DVB-H Service framework.

Essentially, in this Thesis a context-aware DVB-H service software framework was designed and a reference implementation was actualized based on the framework. Realistic and practical operation of the implementation was verified and tested with several scenarios, which confirmed that it is possible and even beneficial to create a recommendation service, whose target content delivery medium is DVB-H.

The developed software framework consists of DVB-H Recommendation Service, DVB-H Client and external services required by DVB-H Recommendation Service to function correctly. DVB-H Recommendation Service is a server-side entity, which is responsible for scheduling the content and, at appropriate moments, initiating the recommendation process, which involves inferring, generating and storing the recommendations. DVB-H Client is a mobile device application, whose main function is to query DVB-H Recommendation Service for new recommendations and, if applicable, display them to the user. The external services of the framework offer DVB-H Recommendation Service information related to 1) who is using the service, 2) the contexts and profiles of the users of the service and 3) the content and its metadata. The reference im-

plementation utilizes several technologies, such as Java Servlet, Java ME and MySQL. These technologies were simple enough for meeting the requirements of the developed software framework.

While the developed software framework and the reference implementation prove it is possible to have a context-aware DVB-H service, there is a substantial amount of work and research to be done in order to overcome the challenges currently afflicting the software. In many cases, the problems faced are related to the lack or complexity of available technologies and methods for utilizing the theoretical concepts and applying them into something practical. The process of designing the developed software framework also could have been more systematic and concentrate more intricately on the major challenges of designing a context-aware software framework. Some of the flaws of the framework could have been avoided by both taking a closer look at the specific requirements of context-aware applications and also by more time and expertise in the related research area.

Nevertheless, the author of this Thesis considers Context-Aware DVB-H Service as novel and functioning approach to the problem of combining DVB-H with the concepts of context-awareness and personalization. In essence, this mobile service framework offers a good basis and original ideas for the developers of context-aware DVB-H-based services to synthesize a framework, which would ultimately lead to the breakthrough of using DVB-H as the prevalent means of delivering relevant content to the users of mobile services.

BIBLIOGRAPHY

1. Zhang, Z. & Zheng, Y. *User Profiles and User Requirements in Mobile Services*. 6th International Conference on Perspectives in Business Information Research, Tampere, Finland, 2007, pp. 170-183.
2. Schilit, B. & Theimer, M. *Disseminating active map information to mobile Hosts*. IEEE Network, 1994, vol. 8(5), pp. 22–32.
3. Gartner. *Gartner's 2010 Hype Cycle Special Report Evaluates Maturity of 1,800 Technologies* [online]. Available from: <http://www.gartner.com/it/page.jsp?id=1447613> (Accessed 16.12.2010).
4. Zhang, D. & Adipat, B. & Mowafi, Y. *User-Centered Context-Aware Mobile Applications - The Next Generation of Personal Mobile Computing*. Communications of the Association for Information Systems, 2009, vol. 24(3).
5. Hiltunen, K-M. & Häkkinen, J. & Tuomela, U. *Subjective Understanding of Context Attributes: A Case Study*. OZCHI '05: Proceedings of the 19th conference of the computer-human interaction special interest group of Australia on Computer-human interaction, 2005.
6. Chevalier, N. *Maximizing Mobile TV Revenues*. Mobile TV World Conference, Rome, Italy, 2007.
7. ABI Research. *Mobile TV Service: Broadcast and Unicast Business Models, Operator Strategies, Devices, and Infrastructure*. Research Report, 2007.
8. DVB Project Office. *DVB Mobile TV: Services, Trials & Pilots* [online]. Available from: <http://www.dvb-h.org/services.htm> (Accessed 31.8.2010).
9. DVB Project Office. *DVB Fact Sheet – April 2009; Broadcasting to Handhelds* [online]. Available from: <http://www.dvb-h.org/PDF/dvb-h-fact-sheet.0409.pdf> (Accessed: 31.8.2010).
10. Hoikkanen, A. *Economics of Wireless Broadcasting over DVB-H Networks*. 5th Annual Wireless Telecommunications Symposium WTS 2006, California, USA, 2006.
11. Dey, A.K. & Häkkinen, J. *Context-Awareness and Mobile Devices*, User interface design and evaluation for mobile technology, 2008, vol. 1, pp. 205-217.
12. Laakko, T. & Goh, K.-H. *CoDe Deliverable 7.3 - CoDe FINAL REPORT*. VTT, 2010.
13. Arjona, A. *Internet Protocol Datacasting - Transparent Interactivity Using Different Communication Channels*. Master's Thesis, Helsinki University of Technology, Finland, 2005.

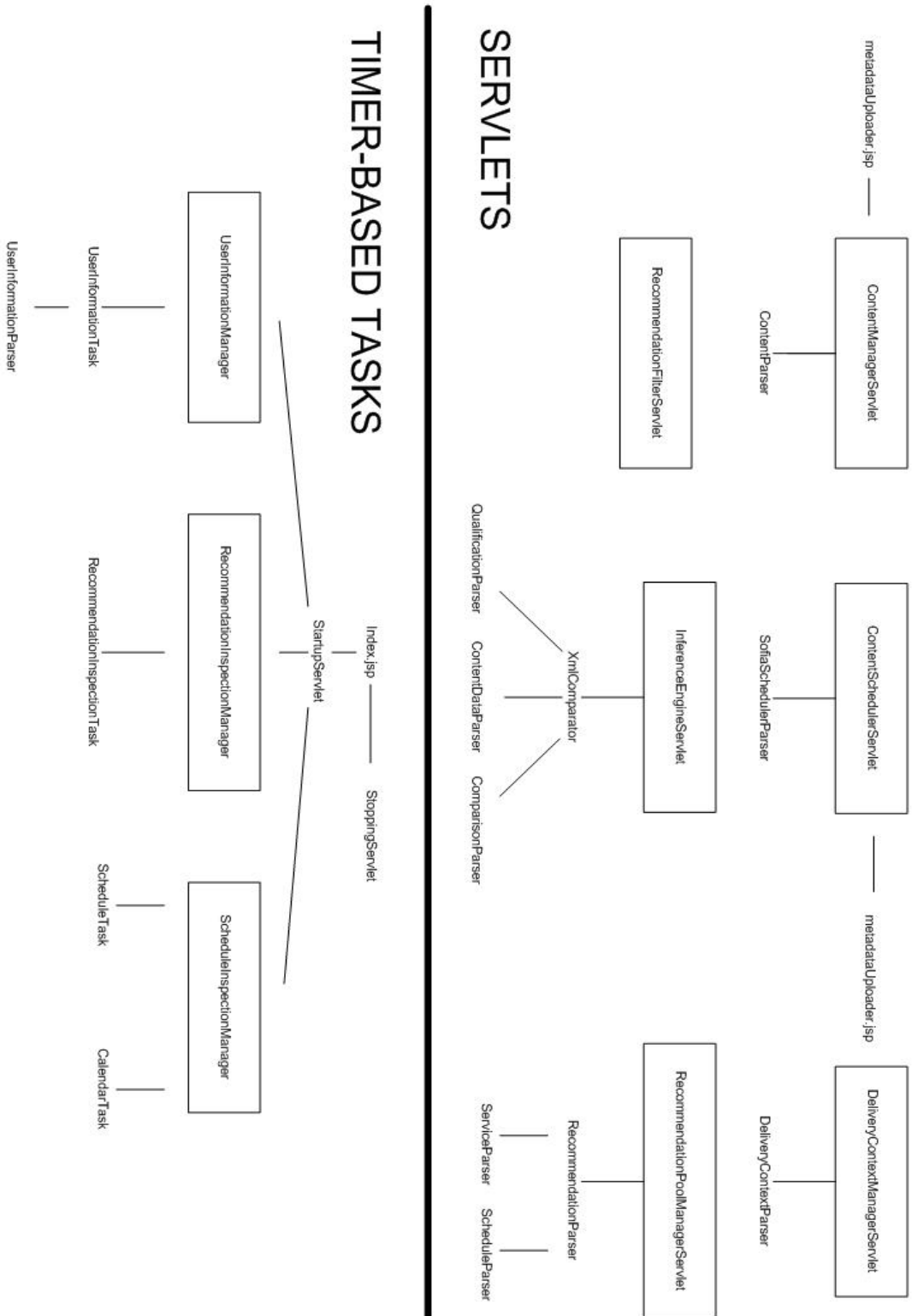
14. Hornsby, A. & Bouzazizi, I. & Defee, I. *Notification Service for DVB-H Mobile Broadcast*. IEEE Wireless Communications, 2010.
15. ETSI TS 102 471 V1.2.1. *Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Electronic Service Guide (ESG)*. 2006.
16. ETSI TS 102 470 V1.1.1. *IP Datacast: Program Specific Information (PSI)/Service Information (SI); Part 1: Datacast over DVB-H*. 2008.
17. ETSI EN 302 304 V1.1.1. *Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H)*. 2004.
18. Abowd, G.D. & Dey, A.K. Towards a Better Understanding of Context and Context-Awareness. Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing (HUC '99), 1999, pp 304–307.
19. Dey, A.K. & Salber, D. & Abowd, G.D. *A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications*. Human-Computer Interaction Journal, 2001, vol. 16(2-4), pp. 97-166.
20. Gustafsson, J. *Hajautettu kehys kontekstietoisten mobiilisovellusten toteuttamiseen*. Master's thesis, University of Helsinki, 2010.
21. Prekop, P. & Burnett M. *Activities, context and ubiquitous computing*. Computer Communications, 2003, vol. 26(1), pp. 1168-1176.
22. Wikipedia. *Ontology (information science)* [online]. Available from: http://en.wikipedia.org/wiki/Ontology_%28information_science%29 (Accessed 3.9.2010).
23. Borst, W.N. *Construction of Engineering Ontologies*, PhD Thesis, University of Twente, Enschede, 1997.
24. Weissenberg, N. & Gartmann, R. & Voisard, A. *An Ontology-Based Approach to Personalized Situation-Aware Mobile Service Supply*. Geoinformatica, 2006, vol. 10(1), pp. 55–90.
25. Strang, T. & Linnhoff-Popien, C. *A Context Modeling Survey*, Proceedings of *Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004, The Sixth International Conference on Ubiquitous Computing*, 2004.
26. Laakko, T. *Context-Aware Web Content Adaptation for Mobile User Agents*. R. Nayak, et al. (Eds.): *Evolution of the Web in Artificial Intelligence Environments*, Studies in Computational Intelligence, Springer-Verlag, 2008, vol. 130, pp. 69–99.
27. Laakko, T. & Gustafsson, J. *CoDe Deliverable 4.2 – Final context and personalisation report*. VTT, 2010.

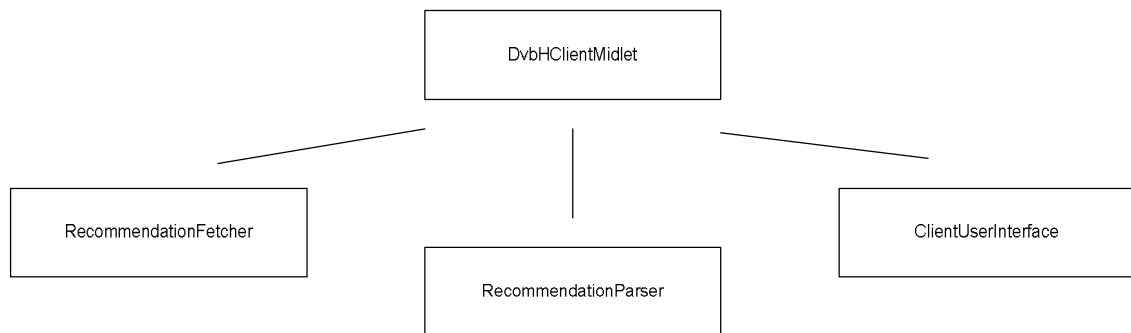
28. Chorianopoulos, K. *Personalized and mobile digital TV applications*, Multimedia tools and applications, Springer Science + Business Media, 2008, vol. 36(1-2):1-10.
29. Fong, S. & Ho, Y & Hang, Y. *Using Genetic Algorithm for Hybrid Modes of Collaborative Filtering in Online Recommenders*. Proceedings of the Eighth International Conference on Hybrid Intelligent Systems, 2008. Pp. 174-179.
30. Various contributors. *Deliverable D16 - Validation of Results and Demonstration* [online]. C-CAST-project, 2010. Available from: http://www.ict-cast.eu/files/C-Cast_D16_revised.pdf (Accessed 15.10.2010).
31. Wikipedia. *Push technology* [online]. Available from: http://en.wikipedia.org/wiki/Push_technology (Accessed 9.9.2010).
32. Wikipedia. *Pull technology* [online]. Available from: http://en.wikipedia.org/wiki/Pull_technology (Accessed 9.9.2010).
33. Vilas, A.F. & Redondo, R.D. & Arias, J.P. & Cabrer, M.R. & Duque, J.G. & Pardavila, R.T. *MyTV 2.0: Semantic Reasoning and Web 2.0 for Mobile TV*. IEEE International Symposium on Consumer Electronics (ISCE 2008), 2008.
34. Fielding, R.T. & Taylor, R.N. *Principled design of the modern Web architecture*. ACM Transactions on Internet Technology (TOIT), 2002, vol. 2(2), pp. 115-150.
35. Pautasso, C. & Zimmermann, O. & Leymann, F. *RESTful Web services vs. "big" Web services: making the right architectural decision*. Proceedings of the 17th international conference on World Wide Web, ACM, Beijing, China, 2008.
36. Scribner, K. & Seely, S. *Effective rest services via .NET*, Boston: Addison-Wesley, ISBN 978-0-321-61325-7, 2009.
37. Coward, D. & Yoshida Y. *Java™ Servlet Specification Version 2.4*. Sun Microsystems, 2003.
38. Zeiger, S. *Servlet Essentials - Chapter 1* [online]. 1999. Available from: <http://www.novocode.com/doc/servlet-essentials/chapter1.html> (Accessed 15.10.2010).
39. Oracle. *Java ME Technology* [online]. Available from: <http://www.oracle.com/technetwork/java/javame/tech/index.html> (Accessed 15.10.2010).
40. Wikipedia. *MySQL* [online]. Available from: <http://en.wikipedia.org/wiki/MySQL> (Accessed 15.10.2010).
41. MySQL. *MySQL 5.1 Reference Manual* [online]. Available from: <http://dev.mysql.com/doc/refman/5.1/en/what-is-mysql.html> (Accessed 15.10.2010).

42. Oracle corporation. *MySQL :: MySQL 5.1 Reference Manual :: 21.3 MySQL Connector/J* [online]. Available from: <http://dev.mysql.com/doc/refman/5.1/en/connector-j.html> (Accessed 18.11.2010).
43. Loebbecke, C. & Huyskens, C. & Järvenpää, S.L. *Adoption of Mobile TV Services Among Early Users: Convergence of Familiar Technologies and Emergence of Technology Induced Paradoxes*. In Proceedings of 7th International Conference on Mobile Business, IEEE Press, 2008, pp. 231-239.
44. Kaasinen, E. *User Acceptance of Mobile Services – Value, Ease of Use, Trust and Ease of Adoption*. VTT, Espoo, Finland, VTT Publications 566, 2005.
45. Oksman, V. & Noppari, E. & Tammela, A. & Mäkinen, M. & Ollikainen, V. *Mobile TV in Everyday Life Contexts - Individual Entertainment or Shared Experiences?*. Proceedings of 5th EuroITV, 2007, pp. 215-225.
46. Oksman, V. & Ollikainen, V. & Noppari, E. & Herrero, C. & Tammela, A. 'Podracing': *Experimenting with mobile TV content consumption and delivery methods*. Multimedia Systems journal, 2008, vol. 14(2), pp. 105-114.
47. Grobel, J. *Mobile Mass Media: A New Age for Consumers, Business, and Society?*. In Groebel, J., Noel, E., Feldmann, V. (eds.) *Mobile Media*. Lawrence Erlbaum Publishers, 2006.
48. Kaasinen, E. & Kulju, M. & Kivinen, T. & Oksman, V. *User Acceptance of Mobile TV Services*, Proceedings of International Conference in Human-Computer Interaction with Mobile Devices and Services (MobileHCI), Bonn, Germany, 2009.
49. Schmidt, A. *Implicit human-computer interaction through context*. Personal and Ubiquitous Computing, 2000, vol. 4(2), pp. 191-199.
50. Dey, A. K. *Understanding and Using Context*. Personal and Ubiquitous Computing Journal, 2001.
51. Nummiahho, A. *Manipulating and Using Context Information in Context-Aware User Interfaces*. Licentiate's thesis, Helsinki University of Technology, 2007.
52. Sofia Digital Ltd. *Sofia Backstage Service Manager for DVB-H* [Online]. Available from: <http://www.sofiadigital.com/products/all-products/146--sofia-backstager-service-manager-for-dvb-h> (Accessed 9.11.2010).

APPENDICES

APPENDIX 1. DVB-H Recommendation Service Java class hierarchy



APPENDIX 2. DVB-H Client Java class hierarchy

APPENDIX 3. Inference logic of DVB-H Recommendation Service

