UNIVERSITÄT
PASSAU
Fakultät für Mathematik und Informatik

Distributed and Multimedia
InformationSystems
Prof. Dr. Harald Kosch

University of Passau
Department of Informatics and Mathematics

Chair of Distributed Information Systems
Prof. Dr. Harald Kosch

Dissertation

Personalized Means of Interacting with Multimedia Content

Günther Hölbling

June the 06th, 2011

1st referee: Prof. Dr. Harald Kosch, University of Passau
2nd referee: Prof. Dr. Maximilian Eibl, Chemnitz University of Technology

Acknowledgments

First of all, I would like to thank my supervisor, Prof. Dr. Harald Kosch, for his extensive and kind supervision, and for the opportunity to take part in his research group. He supported me through all of the highs and lows of writing this work and always found the right words to encourage me to finish this thesis. I am also grateful to Prof. Dr. Maximilian Eibl, who gave me the opportunity to discuss and present my work with him and several members of his research group in an extensive manner. Their many suggestions and pieces of advice have helped me in many ways to complete this work.

This work was further made possible by the support of several people who helped in different phases of its creation. Thanks go out to all colleagues of the Chair of Distributed Information Systems, and especially to Tilmann Rabl, David Coquil, Stella Stars, Mario Döller and Florian Stegmaier for many helpful hints, interesting discussions and valuable proofreading. Thanks also go out to my students Wolfgang Pfnür, Raphael Pigulla, Michael Pleschgatternig and Georg Stattenberger for all their work, and most notably to Andreas Thalhammer for the comprehensive discussions and his support of this work. I also acknowledge the kind help of many supporters who made the creation of our evaluation dataset possible, and Lauren Shaw for many hours of proofreading.

Last but not least, I owe a debt of gratitude to my wife Ines and my two beautiful little children Simon and Sophie. Without their love, support and appreciation it would not have been possible for me to write this thesis.

Abstract

Today the world of multimedia is almost completely device- and content-centered. It focuses it's energy nearly exclusively on technical issues such as computing power, network specifics or content and device characteristics and capabilities. In most multimedia systems, the presentation of multimedia content and the basic controls for playback are main issues. Because of this, a very passive user experience, comparable to that of traditional TV, is most often provided.

In the face of recent developments and changes in the realm of multimedia and mass media, this "traditional" focus seems outdated. The increasing use of multimedia content on mobile devices, along with the continuous growth in the amount and variety of content available, make necessary an urgent re-orientation of this domain. In order to highlight the depth of the increasingly difficult situation faced by users of such systems, it is only logical that these individuals be brought to the center of attention.

In this thesis we consider these trends and developments by applying concepts and mechanisms to multimedia systems that were first introduced in the domain of user-centrism. Central to the concept of user-centrism is that devices should provide users with an easy way to access services and applications. Thus, the current challenge is to combine mobility, additional services and easy access in a single and user-centric approach. This thesis presents a framework for introducing and supporting several of the key concepts of user-centrism in multimedia systems. Additionally, a new definition of a user-centric multimedia framework has been developed and implemented.

To satisfy the user's need for mobility and flexibility, our framework makes possible seamless media and service consumption. The main aim of session mobility is to help people cope with the increasing number of different devices in use. Using a mobile agent system, multimedia sessions can be transfered between different devices in a context-sensitive way. The use of the international standard MPEG-21 guarantees extensibility and the integration of content adaptation mechanisms.

Furthermore, a concept is presented that will allow for individualized and personalized selection and face the need for finding appropriate content. All of which can be done, using this approach, in an easy and intuitive way. Especially in the realm of television, the demand that such systems cater to the need of the audience is constantly growing. Our approach combines content-filtering methods, state-of-the-art classification techniques and mechanisms well known from the area of information retrieval and text mining. These are all utilized for the generation of recommendations in a promising new way. Additionally, concepts from the area of collaborative tagging systems are also used. An extensive experimental evaluation resulted in several interesting findings and proves the applicability

of our approach.

In contrast to the ”lean-back” experience of traditional media consumption, interactive media services offer a solution to make possible the active participation of the audience. Thus, we present a concept which enables the use of interactive media services on mobile devices in a personalized way. Finally, a use case for enriching TV with additional content and services demonstrates the feasibility of this concept.

Zusammenfassung

Die heutige Welt der Medien und der multimedialen Inhalte ist nahezu ausschließlich inhalts- und geräteorientiert. Im Fokus verschiedener Systeme und Entwicklungen stehen oft primär die Art und Weise der Inhaltspräsentation und technische Spezifika, die meist geräteabhängig sind. Die zunehmende Menge und Vielfalt an multimedialen Inhalten und der verstärkte Einsatz von mobilen Geräten machen ein Umdenken bei der Konzeption von Multimedia Systemen und Frameworks dringend notwendig. Statt an eher starren und passiven Konzepten, wie sie aus dem TV Umfeld bekannt sind, festzuhalten, sollte der Nutzer in den Fokus der multimedialen Konzepte rücken. Um dem Nutzer im Umgang mit dieser immer komplexeren und schwierigen Situation zu helfen, ist ein Umdenken im grundlegenden Paradigma des Medienkonsums notwendig. Durch eine Fokussierung auf den Nutzer kann der beschriebenen Situation entgegengewirkt werden.

In der folgenden Arbeit wird auf Konzepte aus dem Bereich Nutzerzentrierung zurückgegriffen, um diese auf den Medienbereich zu übertragen und sie im Sinne einer stärker nutzerspezifischen und nutzerorientierten Ausrichtung einzusetzen. Im Fokus steht hierbei der TV-Bereich, wobei die meisten Konzepte auch auf die allgemeine Mediennutzung übertragbar sind. Im Folgenden wird ein Framework für die Unterstützung der wichtigsten Konzepte der Nutzerzentrierung im Multimedia Bereich vorgestellt.

Um dem Trend zur mobilen Mediennutzung Sorge zu tragen, ermöglicht das vorgestellte Framework die Nutzung von multimedialen Diensten und Inhalten auf und über die Grenzen verschiedener Geräte und Netzwerke hinweg (Session mobility). Durch die Nutzung einer mobilen Agentenplattform in Kombination mit dem MPEG-21 Standard konnte ein neuer und flexibel erweiterbarer Ansatz zur Mobilität von Benutzungssitzungen realisiert werden.

Im Zusammenhang mit der stetig wachsenden Menge an Inhalten und Diensten stellt diese Arbeit ein Konzept zur einfachen und individualisierten Selektion und dem Auffinden von interessanten Inhalten und Diensten in einer kontextspezifischen Weise vor. Hierbei werden Konzepte und Methoden des inhaltsbasierten Filterns, aktuelle Klassifikationsmechanismen und Methoden aus dem Bereich des "Textminings" in neuer Art und Weise in einem Multimedia Empfehlungssystem eingesetzt. Zusätzlich sind Methoden des Web 2.0 in eine als Tag-basierte kollaborative Komponente integriert. In einer umfassenden Evaluation wurde sowohl die Umsetzbarkeit als auch der Mehrwert dieser Komponente demonstriert.

Eine aktivere Beteiligung im Medienkonsum ermöglicht unsere iTV Komponente. Sie unterstützt das Anbieten und die Nutzung von interaktiven Diensten, begleitend zum Medienkonsum, auf mobilen Geräten. Basierend auf einem Szenario zur Anreicherung von TV Sendungen um interaktive Dienste konnte die Umsetzbarkeit dieses Konzepts demonstriert werden.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

According to the studies of the IDC[1], which attempt to measure and to forecast the amounts and types of digital information created and copied around the world, it is predicted that 1,2 trillion GB (1,2 Zettabytes) of digital info will be created in 2010. The study concludes with a forecast, that by 2020 this amount will rise to about 35 Zettabytes. A main factor in this extensive growth is the digitalization of a variety of forms of media ranging from print, voice and radio to televison (TV). It is, however, not only the amount of data that is rising, the quantity and types of different media items offered by a wide variety of sources is also expanding. Consider the popular video platform YouTube[2]. In 2009 more than 200,000 videos were uploaded and published each day on the platform, compared to only 65.000 videos per day in 2007. Recent statistics show that every minute up to 24 hours of video are uploaded[3]. Additionally, in the field of television, digitalization led to an increasing number of channels and as a result, to a further expansion of program choices. Other developments such as interactive TV services, where the user may directely interact with the television content (e.g. participate in a quiz show or gather additional information on news topics), and the integration of internet services into TV-sets, further intensify this growth.

Aside from the massive expansion of the amount of data and media available, the way media is consumed is also changing. In the past, the TV-set and the radio have been central elements of media consumption. Today, mobile devices such as smartphones, tablet PCs, Personal Digital Assistants (PDAs) and notebooks play an increasing role in the media consumption. Young people are especially more and more interested in media consumption via mobile devices (cf. TNS Emnid Medien- und Sozialforschung GmbH - ”Medien to go”[4]).

As a result of these two main trends, users find themselves in the increasingly difficult and unpleasant situation of having to locate interesting and relevant content from among a tremendously growing amount and variety of offers. They must then decide which device is the best (e.g. in terms of performance or resolution), or at least a fitting choice for their

[1] IDC Digital Universe Study - http://www.emc.com/leadership/programs/digital-universe.htm

[2] YouTube – http://www.youtube.com/

[3] ComScore U.S. Online Video Rankings May 2010 – http://www.comscore.com/

[4] TNS Emnid - ”Medien to go” – http://www.radiozentrale.de/site/795.0.html

consumption. Furthermore, barriers often arise because of a variety of issues including the use of different media coder and decoder (codec), different resolution of the devices or different network connections, when users try to consume media with heterogeneous devices on different networks. Currently, the world of multimedia is almost completely device- and content-centered, focused solely on technical issues such as computing power, network specifics or content and device characteristics and capabilities. In light of the current unpleasant situation, we propose that the user be brought to the center of attention of multimedia systems, in order to better reflect current trends. This demand is commonly referred to as ”user-centrism.” It is a well known concept in different areas of application. Very early adoptions of user-centrism can be found in User-Centered Design (UCD), which has its roots in the late 1980s. In UCD, the user's needs, demands and characteristics are considered right from the beginning of the design process of a system. UCD is closely related to the terms ”usability” and ”usability engineering.” It focuses on the way the ”real” user is going to use a specific system or product, with respect to the products usage and the user context. According to [Dey01] the context can be defined as follows: ”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 an user and an application, including the user and applications themselves.” Furthermore, user-centrism is also a central concept in the area of ubiquitous computing. In [Cho06] the user-centric point of view is adopted to the domain of information retrieval. The authors propose a system for the visualization of multimedia query results for mobile devices where two different user specific sorting criteria, corresponding to the user's query, are used for enhancing the results presented. Additionally, this approach also supports service migration between different devices, for the proposed retrieval use case.

According to the authors of [Man02], user-centrism requires applications and systems to:

1. be portable and adapt to new locations as the user moves
2. adapt according to changes in available resources (it may imply data format transformation, or internal application composition, or both)
3. provide mechanisms to allow users to configure the application according to their personal preferences

Following this definition, aspects like the location of the user, available resources and his or her personal preferences seem to be key concepts to providing user-centrism. User-centered applications are generally defined as applications that are bound to a specific user, and that react and adapt in a resource-aware manner. In the following, we will focus on user-centrism in multimedia systems.

1.1 User-Centric Multimedia

The concept of user-centrism is also considered to be a way of keeping track of the recent developments in multimedia. Thus, it is important to identify and define proper requirements for the application of user-centrism to multimedia systems. In the area of networked multimedia systems, Reiterer et al. [Rei08b] define the following requirements:

- *Easy access to available content repositories:* This requirement focuses specifically on the integration of content, provided by different resources, and an common way of usage.
- *Context awareness:* The context information (location, available resources, etc.) is used to provide seamless consumption of multimedia content e.g. selecting a rendering device based on its location to the user.
- *Content adaptation:* The content is tailored and adapted to the characteristics of a specific target device and the properties of the network for a satisfying media experience.
- *Session migration:* Because it has become common for several multimedia devices to be available in a home environment, the user should be able to take media sessions with him by migrating them from one device to another.

A common definition of ”context awareness” is given in [Dey01] as follows: ”A system is context-aware, if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.”

In this thesis, a user-centric application is commonly understood as an application that is directly tied to the user, instead of to a specific device, resource, service or content. By taking the different definitions, requirements and concepts into account, we have created our own, application-oriented, specification. A user-centric multimedia system must be able to provide:

- a seamless consumption of contents and services on and transferable to different devices (session mobility)
- device and context aware adaptation, selection and composition of content, format and services
- a comprehensive personalization based on user preferences and settings (e.g. assisted selection of content, recommendation generation, personal channel, personal interface, etc.)

This definition is based on the idea that the need for user-centrism has been extended by the demand for comprehensive personalization. In our opinion, this is a key feature for user to cope with the rising amount of available contents. Otherwise, without proper selection support, the user will not be able to find relevant multimedia content in the future.

1.2 Framework Overview

Facing the challenges of user-centric multimedia systems, as described in our definition, several areas of interest for this work have been identified. Figure 1.1 shows the concept of our user-centric multimedia framework. Within this framework the following questions are addressed:

1. *How do you provide a framework for seamless media and service consumption?*

2. *How do you support a user in a personalized way, especially in the selection of content?*

3. *How do you enhance the audience's experience using additional services? (e.g. intelligent content selection, lotteries, information services, etc.)*

Note that although most concepts and mechanisms presented here can be easily transfered and adapted to multimedia systems in general, this thesis focuses on the field of TV, as present in a typical home environment. A central component of our multimedia framework is a typical TV set-top box (STB). It is used to offer, besides basic TV-related functions, such as tuning to a specific channel or recording a program, additional services to all users of a home network. TV metadata is heavily used by these services and therefore, is at the foundation of our system. TV content is usually provided and accessed by one of the following sources: Cable, terrestrial antenna, satellite dish or the internet. Generally, various kinds of devices such as PCs, notebooks, PDA's or smartphones can be used with our system. Based on an ad-hoc service architecture, namely Universal Plug and Play (UPnP), our services may be easily discovered and used in a way which is also accessible for inexperienced users. As user's may employ different devices for service consumption, a mechanism for handling session mobility (cf. chapter 5) has been introduced. Based on a mobile agent system, namely the Java Agent Development Framework (JADE), users are able to "take their service session with them without interruption (seamless)." Session migration between different devices is offered in an almost automatic manner. Besides session mobility and easy access, which are enabled by our framework, main elements of the approach are realized in the form of two services:

- iTV service: The interactive TV (iTV) service offers additional TV related applications in a synchronized way. Evidence of this concept can be seen in interactive game

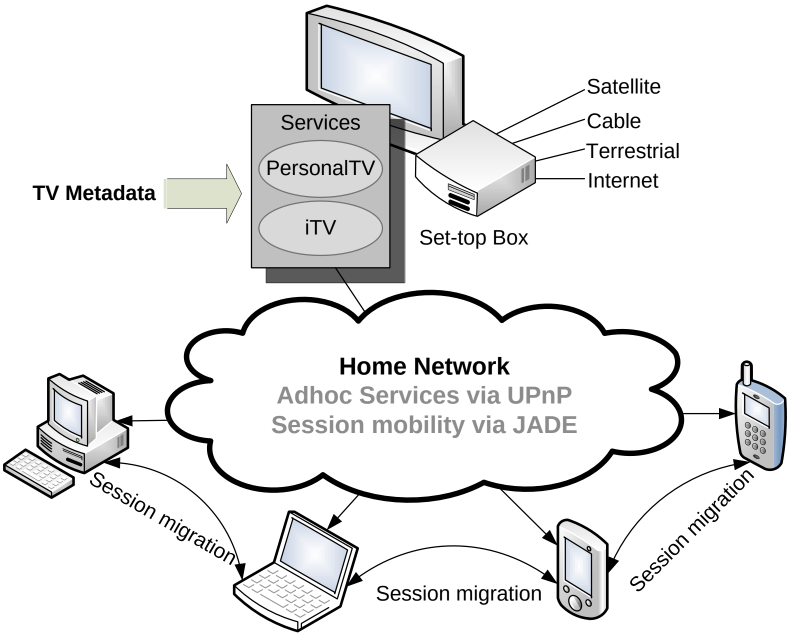


Figure 1.1: Overview of the concept of our user-centric multimedia framework.

shows and enhanced news tickers that have been implemented as an iTV service in our framework.

- PersonalTV service: To allow for personalized and individualized content selection support, the personalTV service is offered. It has been implemented as a typical recommendation system for TV content. In general, it is not limited to TV content and can also be used with arbitrary types of multimedia as long as adequately descriptive metadata are available.

For a detailed discussion of the iTV service, the reader is referred to chapter 5 and for a discussion of the personalTV service, to chapter 6. In addition to the services and mechanisms realized in this work, our framework is well suited for testing new layouts for remote-controls and new interaction concepts for the TV domain. Based on the advanced capabilities of different mobile devices, a variety of innovative concepts may be easily tested and evaluated in our experimental environment.

1.3 Structure of this Work

In this section we will take a closer look at the organization and the structure of this thesis.

First, chapters 2 to 4 introduce several fundamental concepts and mechanisms used within this work. These chapters lay the technical foundation and provide a common understanding of the different concepts referred to in the implementation of our prototypical applications and framework.

In chapter 2 we provide specific details about the TV domain. Its first part (cf. section 2.1) discusses the emergence of iTV and examines its technical foundation and available standards. Additionally, this chapter covers metadata formats and standards widely used in the multimedia and TV domain. A structured comparison of these standards provides the basis for the selection of a specific standard that has been used in our approach. Chapter 3 introduces basic concepts of the ideas of natural language processing and text mining. While focusing on the proper splitting of natural text and its morphological, syntactical and semantical augmentation, this chapter discusses concepts and techniques used in the first processing steps of TV program descriptions in our system. The principles of recommender systems are introduced in chapter 4. In addition to a detailed categorization of such systems, the major conceptual approaches used for recommendation generation are discussed as well.

Chapters 5 and 6 focus on the main conceptual and theoretical contributions, and cover several implementation specific aspects of this work.

In chapter 5 we concentrate on seamless media and service consumption, and on offering additional TV related services. In the first part of this chapter, our prototypical framework for providing session mobility in mobile environments is presented. It makes use of a mobile agent-based approach to remove a service's dependence on a specific device, and attach it instead to the user. In the second part, our approach to providing the TV audience with additional iTV services on mobile devices is presented. At this point the principles of user-centrism are brought into focus by allowing access to these services in an easy and almost configuration-free way. Chapter 6 presents the main contribution of this work. It introduces our personalization approach, which aims to aid the user in their selection of content. First, specific techniques used, such as different content filtering approaches, methods from the pattern recognition domain, as well as our approach to build an ”intelligent” text splitting (tokenization) mechanism, are discussed in detail. Accordingly, a comprehensive description of our personalization system is presented, including its two main components, the content-based and the collaborative media recommender. A detailed evaluation of all system components concludes this chapter.

Finally, in chapter 7 a short summary and an extensive discussion of future prospects and developments is given. Additionally, recent trend-setting developments and research activities are examined in this chapter.

CHAPTER 2

Interactivity in TV and TV Metadata

Metadata is essential in many domains, stretching across all levels of digital content presentation and archiving. It is often defined as structured data which describes different characteristics of content or, more loosely, as data about data. For instance, a simple metadata record for a book may provide information concerning the title, publisher, authors, ISBN and much more. In general, there is no explicit difference between metadata and data. It mainly depends on the point of view and the application domain, as to whether a specific piece of data is interpreted as metadata or data. Metadata is a key element in enabling and improving searchability and modeling relationships between individual pieces of content. Furthermore, it also plays an important role in the realm of multimedia and TV, particularly in navigating through content collections, and in organizing and finding content. Referring to the definition of user-centrism in multimedia, metadata is needed to allow for content adaptation, personalization and much more. Generally, it also provides the foundation for offering additional and interactive applications in the realm of TV.

The following chapter is structured as follows: In the first section we will discuss all topics related to interactivity in TV, ranging from the definition and technical aspects, to levels of interactivity. Additionally, the historical point of view of different middleware standards used to enable interactive TV services and applications are discussed. Section 2.2 covers the role of metadata throughout the audiovisual media production process and provide details on different, widely used TV metadata standards.

2.1 TV Interactivity

The term ”interactive television” (iTV or ITV) is used for television systems in which the audience is able to interact with TV content. It is, however, not usually used to imply that the viewer is able to change the storyline of a program. Instead, he may, for instance, participate in a quiz show, gather additional information on news topics or directly buy a product presented in a commercial. In addition to this, Electronic Program Guides (EPGs), Video on Demand (VoD) portals or Telelearning can be made possible with iTV Systems.

Interactivity in television is not as new as one might think. The roots of iTV go back to the 1960’s, where certain game and quiz shows allowed viewers to call in and participate in the show. In 1974, the teletext was developed in the United Kingdom. It was the first form of additional content delivered with the broadcast programs. At the Internationale Funkausstellung (IFA) 1979 in Berlin, the Tele-Dialog system was presented. It was a televoting system which allowed viewers to participate in polls for TV shows by calling specific telephone numbers. In the mid 1990s more advanced forms of interactivity appeared on the media landscape. One of the first experimental trials was the Full Service Network by Time Warner, which was launched in December 1994. This trail provided several interactive services, like video-on-demand, a program guide, video games and home-shopping to customers. Unfortunately, within a period of 18 months, only 65 people subscribed to this service, and as a result, it was closed. However, with the ongoing digitalization of television, interactivity has become an interesting topic once again, and many broadcasters have tried to enrich their content with interactive services.

In the following section, we begin with a categorization of interactivity in TV (section 2.1.1). iTV systems are complex systems that involve a long chain of successive processes, stretching from the broadcaster to viewers. Section 2.1.2 gives a short overview of the main components of such systems, particularly on the consumer side. One of the most interesting and important components that enable the use of iTV applications are the middleware and iTV standards which are discussed in section 2.1.3. The last section focuses on the interrelation between different standards and their future developments.

2.1.1 Levels of Interactivity

Interactive applications have a widespread diversity of user interfaces and resource requirements, but also offer different levels of interactivity. In this section we will introduce seven levels, and for each level we will give some representative interactive applications. Our categorization is based on [Ruh97].

Level 1 - Basic TV: Interaction on this level is defined as basic functions for watching TV, such as switching channels and powering the TV set on and off.

Level 2 - Call-In-TV: At this level, the interaction between the audience and the broadcaster is established by using techniques such as telephone calls or short message service (SMS). Examples of such TV shows are music programs where viewers my choose the next music clip or televoting shows where viewers can vote for their favored candidate.

Level 3 - Parallel TV: Parallel TV introduces alternative content on multiple channels. The viewer is able to change the way he consumes a broadcast program. Popular examples of broadcast programs on this level are multilingual audio channels or subtitles. Another form of parallel TV are shows with different camera angles, perhaps most well known from auto racing programs. A very special form of parallel TV is made up of movies that show the perspective of different characters on different channels.

Level 4 - Additive TV: This level is also known as ”enhanced TV.” In addition to the TV program, further content is also broadcast. The content can either offer basic information or more advanced services. A well-established service is teletext. Applications like EPG or synchronized program-related services are advanced examples of this. Generally, a return channel is not needed for applications on this level.

Level 5 - Service on Demand: The ”Media on Demand” level enables the viewer to consume programs, independent of the TV schedule. This level includes video on demand (VoD), upgrade services and other services that are provided when a customer requests them. The interaction between the user and the service provider requires a return channel. In TV environment, near-VoD is also frequently used. Near-VoD uses several channels, on which multiple copies of a program are broadcast in short intervals.

Level 6 - Communicative TV: For TV programs, content from other sources, such as the internet can be accessed in addition to broadcast content. Services that originate in the PC domain can also be used in combination with TV. At this level, TV may additionally be enriched by community functions: chats, online games, social networks or email. Another option is user-generated content that can be uploaded. As a result, user- or community-generated programs become possible at this level (cf. our tagging based recommendation approach in section 6.4)

Level 7 - Fully Interactive TV: The most enhanced level of interactivity enables the user to create his or her individual storyline for a program. A program on this level can be understood as a kind of video game in which the user affects the proceeding of the program. The program can also be affected automatically based on the personal profiles of different users, and as a result may include personalized commercials as well as personalized movies.

Today, most iTV applications can be assigned to the level 4, 5 or 6. User-generated content, as mentioned on level 6, is also increasingly present in the world of iTV. Applications on level 7 are still in their infancy, although several approaches for personalized commercials are in work (e.g. the selection of advertisements that are relevant to a specific person[1]).

2.1.2 Basic Technologies for iTV

Before discussing the different iTV standards, we will give a brief introduction into the basic technology of iTV.

Digital TV Standards

The term ”Digital TV” (DTV) or ”Digital Video Broadcasting” (DVB) ’is used to describe the transmission of digitized audio, video and auxiliary data. As for analog broadcasting, the digital TV market is fragmented. There are various standards developed by different organizations all over the world. This fragmentation is further intensified by the way the DTV standards are further adapted to accommodate the requirements of different transmission channels used for broadcasting. For that reason, there are different standards for terrestrial, satellite, cable and mobile TV. In Europe, the DTV standards of the Digital Video Broadcasting Project[2] (DVB) are used. Other major players in the development

[1] United States Patent Application: 20070174117; Advertising that is relevant to a person by the Microsoft Corporation

[2] Digital Video Broadcasting Project - http://www.dvb.org/

of DTV standards are the Advanced Television Systems Committee[1] (ATSC) and the CableLabs[2] in the US, and the Association of Radio Industries and Businesses[3] (ARIB) in Japan.

Set-top Boxes and Media Centers

A set-top box (STB) is a media device that forms a link between a TV-set and an external source. The source is usually one of the following: a cable, terrestrial antenna, a satellite dish or, in IPTV systems, the internet. According to more flexible definitions, all forms of electronic devices that are connected to a TV-set are called set-top boxes. The term ”set-top box” is, however, generally used to describe typical Consumer Electronic (CE) devices and stems from the fact that STBs are usually placed above the TV-set. STBs have multiple purposes, ranging from simple signal conversion (e.g. digital receiver) to personal video recorder (PVR) and interactive media center functions. Figure 2.1 shows a rough schematic overview of a typical STB architecture. In general, components such as memory, processor and graphics vary widely between different STBs and are therefore not further discussed. In the following, we will briefly discuss the main components of STB architecture.

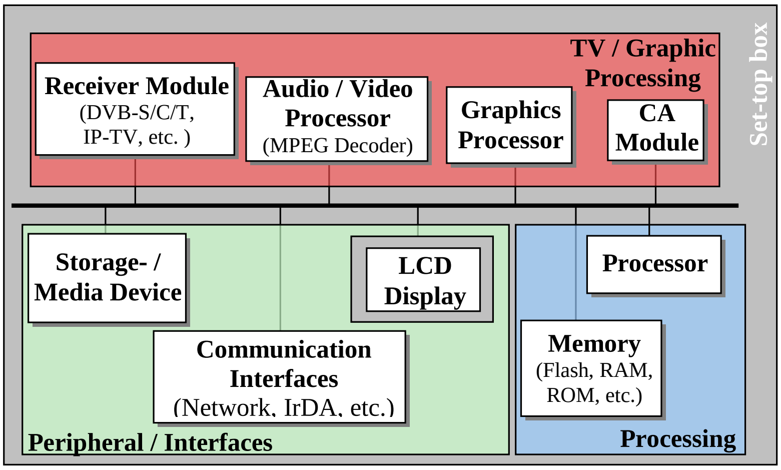

TV / Graphic Processing
Set-top box
Receiver Module (DVB-S/C/T, IP-TV, etc.)
Audio / Video Processor (MPEG Decoder)
Graphics Processor
CA Module
Storage- / Media Device
LCD Display
Communication Interfaces (Network, IrDA, etc.)
Peripheral / Interfaces
Processor
Memory (Flash, RAM, ROM, etc.)
Processing

Figure 2.1: Schematic architecture of a set-top box.

- Receiver Module: Every STB typically needs one or more receiver modules to receive, demodulate and prepare the broadcast for further processing. Depending on the specific transmission media in use, different types of receiver components are used. Common examples are DVB-T/T2 for terrestrial, DVB-C/C2 for Cable and DVB-S/S2 for satellite transmission. To support the recording and/or the watching of different channels simultaneously, multiple receiver and tuner units are needed to tune into the specific frequencies of the programs in the signal.

[1] Advanced Television Systems Committee (ATSC) - http://www.atsc.org/
[2] CableLabs - http://www.cablelabs.com/
[3] The Association of Radio Industries and Businesses (ARIB) - http://www.arib.or.jp/english/

- Audio / Video Processor: Audio and video coding standards play a major role in the world of DTV. The decoding of the video stream is done in software or accomplished in hardware. The encoding of video and audio and, as a result, the reduction of the amount of data enabled the introduction of digital TV. An important standard which is used throughout the DTV domain is the MPEG-2 standard, as defined by the Moving Picture Experts Group[1]. Besides MPEG-2, in several recent DTV standards, MPEG-4 Part 10 Advanced Video Codec (MPEG-4 AVC or H.264) is also used for video coding. In spite of the use of different video and audio encoding standards, the system part of MPEG-2 is of considerable importance for most DTV standards. It describes the combination of multiple encoded audio and video streams and auxiliary data into a single bit stream. This process is called multiplexing. Figure 2.2 shows the multiplexing step in detail. Audio, video and auxiliary data are also called elementary streams (ES). Each ES is split into small packets by the packetizer, which facilitates the processing of the data. After packetizing, each stream is composed of small packets - the packetized elementary streams (PES). The result of the multiplexing step is a bit stream that can be a program stream (PS) or a transport stream (TS). The PS is optimized for use in error-free environments, and provides only one time base for all combined streams. The PES is mainly used for DVDs. The TS is optimized for error-prone environments and is capable of carrying multiple time bases. For that reason, it is used for multiplexed streams in many DTV standards. Different time bases allow for the carrying of multiple TV channels with multiple programs. Furthermore, the multiplex is used to transport additional data and iTV applications in a single stream.

- CA module: The conditional access (CA) module is used to get access to encrypted and scrambled TV content which is generally provided by Pay-TV-provider. The module is similar to the well known Personal Computer Memory Card International

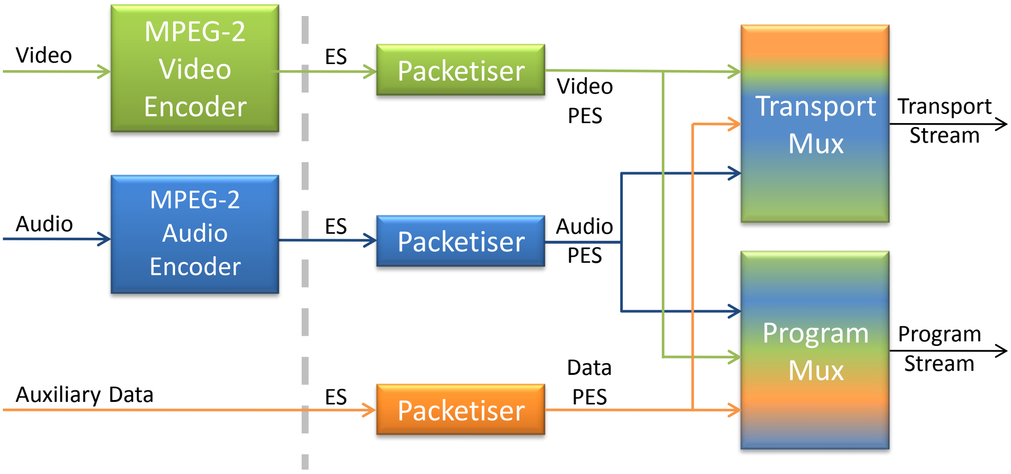


Figure 2.2: Multiplexing step of MPEG-2 (see [Rei04, fig. 5.1]).

[1] The Moving Picture Experts Group (MPEG) - http://mpeg.chiariglione.org/

Association (PCMCIA) card concept. It offers a decryption cipher and software to make possible the computation of decryption keys provided in the received data stream. This key is used to decrypt and descramble the TV content. Typically, these modules offer additional smart card interfaces, in which a smart card is used to identify valid subscribers for specific services. The CA module can be directly integrated into the STBs hardware or created as a removable module. For the insertion of a removable CA module, most STBs offer a Common Interface (CI)[1] or the newer expansion, Common Interface Plus (CI+)[2].

- Storage- / Media Devices: Many STBs include storage and media devices, such as hard disks, DVD or Blu-ray devices. Besides playback functions, storage devices are frequently used to make possible recording and time-shifting functions.

- Communication Interfaces: Communication in STBs is supported by different components and techniques, such as Bluetooth, Wifi, Irda and LAN. For IPTV in particular, the communication interface becomes the main source for TV content. For additional services and interactive TV (cf. section 2.1.3), these components are used to provide a return channel. Furthermore, the integration of STBs into home networks for sharing images, audio, videos, etc. is enabled by these interfaces.

Besides the typical STBs which originate in the consumer electronic domain, media center or Home Theater PCs (HTPC) are gaining more and more relevance in DTV. The convergence of these two, formerly separate device classes is also obvious. Standard PC components are often used in STBs, whereas media center PCs are more and more designed like typical CE devices. As a result, we are experiencing a smooth transition between these two domains. Nevertheless, on the software level, this transition is still in its infancy. Media center PCs offer a wide variety of different software solutions for enabling typical STB functions. These solutions range from proprietary software, such as Windows Media Center (WMC)[3] from Microsoft and Nero MediaHome[4] made by the Nero AG, to open source projects such as Freevo[5], the Video Disk Recorder[6] and MediaPortal[7]. By contrast, the realm of CE STBs is still dominated by different producer specific systems and several standardized middleware platforms. Interactive TV features in particular are almost exclusively supported by STB specific software, which is discussed in detail in section 2.1.3.

2.1.3 Middleware Platforms

In this section we will provide an overview of several open iTV standards and middleware platform specifications. This overview is not exhaustive, but covers the major open

[1] CI is specified by DVB and a ETSI Standard since 1999 (ETSI TS 101 699)

[2] CI+ is specified by the CI Plus LLP – http://www.ci-plus.com/

[3] Windows Media Center – http://windows.microsoft.com/de-AT/windows7/products/features/windows-media-center

[4] MediaHome – http://www.nero.com/deu/mediahome4-introduction.html

[5] Freevo – http://freevo.sourceforge.net/

[6] VDR – http://www.tvdr.de/

[7] MediaPortal – http://www.team-mediaportal.de/

standards. First, we will introduce basic specifications and technologies, such as the DAVIC standards and different parts of the Java platform widely used in the different middlewares. Accordingly, different iTV standards are discussed. This section is concluded by a discussion of the history and future of iTV standards and their interrelationship.

Digital Audio-Video Council (DAVIC)

The Digital Audio-Video Council[1] (DAVIC) was founded in 1994 and completed its work in 1999. The DAVIC standards focused on providing end-to-end interoperability of interactive digital audio-visual applications and services [Dig94]. Since there were many companies involved in DAVIC and the iTV domain, a major goal was to keep the specifications simple and maximize interoperability of applications and services by specifying open interfaces and protocols. Thus, existing standards were used whenever possible. For instance, for multimedia information delivery, the format specified in MHEG-5 was used. The DAVIC standards versions 1.0 (11 parts) - 1.4 (14 parts) cover all areas of commercial interactive multimedia experience, such as definition of service provider -, delivery system -, and service consumer system architecture and interfaces. They also address high, middle and lower layer protocols and physical interfaces. Version 1.5 additionally specified five parts that focus on IP-based audio-visual services. After 5 years, the DAVIC work was completed. Some concepts and parts of DAVIC were adopted and further developed by the TV-Anytime[2] organization. Even today, major parts of the DAVIC specifications are referenced and used in different iTV standards.

Java TV

The Java TV API[3] is an extension of the Java platform. It provides functions for using Java to control and run applications on TV receivers, such as set-top boxes. The main purpose of this extension is to combine the platform independency of Java applications with a set of functions recommended for an iTV platform, as offered by typical TV-specific libraries. Furthermore, Java TV applications are independent of the underlying broadcast network technology. The Java Virtual Machine (JVM) resides in the set-top box and allows for local execution of the applications, which are usually embedded within the broadcast content. Set-top boxes are often very resource-constrained devices. For that reason, the PersonalJava application environment, which is optimized for such devices, is used. PersonalJava offers a subset of APIs introduced by the Java Standard Edition (JSE). Applications using PersonalJava are fully compliant with the JSE. There are several packages in the PersonalJava application environment that are frequently used by Java TV applications. The *java.io* package provides functions for input/output operations. It is used for file-based operations, such as file system access (local and remote) and for stream-based operations such as broadcast data access. The *java.net* package is used for IP-based network access. These functions are often used to provide return channels or for accessing IP data in the MPEG TS. Another important feature is security. To address this need, Java TV makes use of the JDK 1.2 security model, which allows operators to

[1] The Digital Audio-Video Council (DAVIC) - http://www.davic.org/
[2] The TV-Anytime Forum - http://www.tv-anytime.org/
[3] The Java TV API - http://java.sun.com/products/javatv/

define their own security model or policy. Of utmost importance for an iTV system, in terms of security, are issues like the conditional access sub-system, secured communication and the secured execution of code in the JVM. Based on the graphics toolkit, the abstract window toolkit (AWT), user interfaces (UI) can be build. AWT brings with it a set of basic UI components.

As follows, several important aspects and functions of Java TV will be explained [Cal00]:

- Service and Service Information (*javax.tv.service*): A service is often used as a synonym for ”TV channel.” In Java TV, a service is handled as a single unit. It represents a bundle of content (audio, video and data) that can be selected by a user. Service information (SI) describes the content and characteristics of a service. This information can be offered in several formats, depending on which standard is used. Two possible options are DVB-SI and ATSC A56. Java TV offers a common API for accessing service information. Services are composed of several service components. A service component represents one element of a service, such as a video or a Java application. Besides these functions, which are common to all services, more specialized features are also available. The *navigation* sub package provides functions that make it possible to navigate through the existing services and to request detailed information about services and their components. The *guide* sub package provides APIs for EPG. Basic EPG data such as program schedules, program events, and rating information (e.g. parental control information) are also included. The *transport* sub package offers additional information about the transport mechanism used (e.g. MPEG-2 TS). The *selection* sub package provides mechanisms to select discovered services for the presentation. The service context, represented by the *ServiceContext* Class, provides an easy way to control the service and its presentation. In general, the selection of a specific service context, determines the presentation of the service and its components. For example, a selection may cause the receiver to tune to a desired service, demultiplex the necessary service components, present the included audio and video and launch related applications. A service context may exist in one of the following four states - ”Not Presenting,” ”Presentation Pending,” ”Presenting,” ”Destroyed.” Although the number of simultaneous ServiceContext objects is not restricted by any particular specification, a limitation is often imposed by resource constraints.

- JMF: The Java Media Framework[1] (JMF), although not part of the Java TV API, is very important for Java TV. It provides the foundation for management and control of time-based media, such as video and audio, in Java TV. JMF offers a player component, including a GUI for playback of video and audio streams, which aids in the integration and flexible placement of the presentation. In the Java TV API, only controls for video size and positioning (*AWTVideoSizeControl*) and for media selection (*MediaSelectionControl*) have been specified. However, other controls may also be implemented. Moreover, a set of controls, such as a *GainControl* for manipulating audio signal gain, is provided by JMF. Additionally, several useful controls were defined in DAVIC 1.4. Furthermore, JMF provides the foundation for

[1] The Java Media Framework (JMF) - http://java.sun.com/products/java-media/jmf/

a synchronized presentation of time-based media components using an internal clock mechanism.

- Broadcast Data API: The Java TV API provides access to different kinds of broadcast data. Broadcast data is transmitted simultaneously to the video and audio components embedded in the television broadcast signal. The first kind of broadcast data is the broadcast file system. For transmission, broadcast carousel mechanisms are used. In a broadcast carousel, all files are repeatedly transmitted in a cyclic way. The data access in Java TV is modeled after the access to a conventional read-only file system with high access latency. Predominant protocols in the area of broadcast file systems are the Digital Storage Media Command and Control (DSM-CC) data carousel protocol, well known from the teletext service, and the DSM-CC object carousel protocol [Int98b]. DSM-CC is an extension of the MPEG-2 standard. The other two kinds of broadcast data are IP datagrams and streaming data. IP datagrams, unicast and multicast, are accessed via the conventional functions of the *java.net* package. Streaming data is extracted and accessed by the use of JMF.

- Application Life Cycle: Java applications for digital receivers using the Java TV API are called Xlets. Xlets are similar in concept to Java applets. In contrast to ordinary Java applications, Xlets must share the JVM with other Xlets, like applets do. As a result, Xlets have a special application life cycle model and a component, the application manager, that controls and manages their life cycle. There are four states defined in the life cycle of an Xlet. Table 2.1 provides further details on the different states.

  Xlets are optimized for the use on TV receivers. An Xlet also has an associated context, the *XletContext*. This context is an interface between the Xlet and its environment, similar to the *AppletContext* for applets. This interface allows the Xlet to discover information about its environment and to inform its environment about its state changes.

State	**Description**
Loaded	The loading step includes the call of the Xlet's constructor and may include limited initialization. The construction of the new Xlet is the business of the application manager. If an exception occurs in the loading step, the Xlet is destroyed, otherwise the *Loaded* state is entered. This state is only entered once in the Xlet's life cycle.
Paused	This state may be reached from the states *Active* and *Loaded*. It indicates that the Xlet is idle. Before entering this state, all shared resources held by the Xlet must be freed. By calling the *notifyPaused()* method, the application manager is notified about the state transition.
Active	This state may only be reached when the application manager has indicated that the Xlet has the permission to run. Active means that the Xlet is running and currently offering its service.
Destroyed	The *Destroyed* state may be reached from every other state. This state does not allow the transition back to another state. By calling the *notifyDestroyed()* method, the application manager is notified about the state transition.

Table 2.1: Xlet life cycle states.

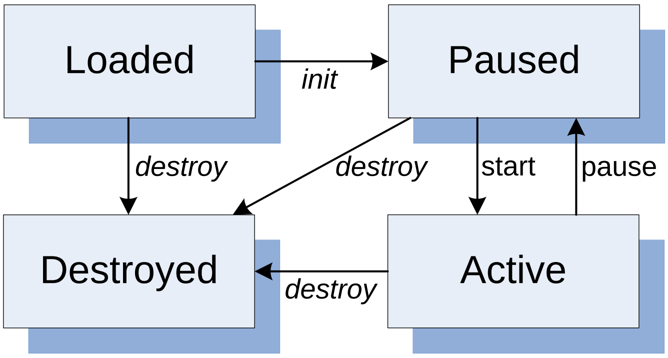


Figure 2.3: Schematic visualization of the Xlet life cycle states.

Java TV provides many interesting concepts for iTV systems. The application model in particular, introduced in Java TV, is used in most major iTV standards. Colloquially speaking, the Xlet concept paved the way for interoperable iTV applications.

Multimedia and Hypermedia Information Coding Expert Group (MHEG)

The Multimedia and Hypermedia Information Coding Expert Group[1] (MHEG), a subgroup of the International Organization for Standardization[2] (ISO), published the MHEG standard in 1997. It was designed as an equivalent to HTML for multimedia presentations [MB95]. Accordingly, the aim of the group was to describe the interrelation between different parts of a multimedia presentation and to provide a common interchange format. The standard initially consisted of five parts, and three parts were added later on:

MHEG-1 - MHEG object representation - base notation (ASN.1) [Int97a]: The first part of the standard defines the encoding of MHEG presentations in the Abstract Syntax Notation One (ASN.1). A central aim of the design was to build a generic standard. As such, it contains no specification about the application area or target platform. MHEG follows an object-oriented approach. An overview of the MHEG class hierarchy can be found in figure 2.4. Media elements of a presentation, such as text, audio and video, are represented by *Content* objects. These may contain information about the media elements, such as spatial and temporal attributes, as well as information about the actual content or solely a reference. *Action*, *Link*, and *Script* objects are used to describe behaviors. Simple objects can be grouped to *Composite* objects. The logic behind this is that objects that are needed together, should be arranged as bundled objects. Limited user interactivity is provided by the *Selection* and *Modification* class. The *Selection* class enables the user to select an item from a predefined set, such as a drop-down menu, while the *Modification* class provides free user input. For a detailed discussion of the remaining classes, the reader is referred to the MHEG standard [Int97a].

[1] Multimedia and Hypermedia Information Coding Expert Group (MHEG) - http://www.mheg.org
[2] International Organization for Standardization (ISO) - http://www.iso.org

- *MH-Object*
  - *Behaviour*
    - Action
    - Link
    - Script
  - *Component*
    - Content
    - *Interaction*
      - Selection
      - Modification
    - Composite
  - Descriptor
  - *Macro*
    - Macro Definition
    - Macro Use

Figure 2.4: Main Classes of the Multimedia and Hypermedia Information Coding Expert Group -1 Class Hierarchy [Rog93].

MHEG-2 - MHEG object representation - alternate notation (SGML): Should have provided an encoding based on the Standard Generalized Markup Language (SGML) instead of ASN.1, but was never finished [Rog93].

MHEG-3 - MHEG script interchange representation[Int97b]: The third part of the MHEG standard defines an encoding model for scripting dedicated to a virtual machine. Because interactive elements of MHEG-1 are very limited, this part features advanced interactive operations. The encoding is based on the Interface Definition Language of CORBA and is known as the Script Interchange Representation [Eur94]. As for the other standard parts, there were no specifications made about the concrete execution environment. The encoding mainly represents an intermediate presentation for the platform-independent exchange of "scriptware." This representation is also suitable as target object code for any script compiler. [Rut96].

MHEG-4 - MHEG Registration Procedure [Int96]: MHEG-4 describes the procedure to register identifiers for objects. For example, those for data formats and script types.

MHEG-5 - Support for base-level Interactive Applications [Int97c]: In MHEG-1 and MHEG-3, many features were too complicated for the technology of their release time. In order to overcome this problem and to support heavily resource-constrained systems, MHEG-5 was developed. Although MHEG-5 is a simplification of MHEG-1, there are too many differences between the two versions to be compatible. The class hierarchy itself is quite different. For a better adaptation to the interactive television domain, the naming was also changed. An MHEG-5 application consists of *Scenes* that are composed of *Ingredients*.

MHEG-6 - Support for Enhanced Interactive Applications[Int98a]: Although MHEG-1 had already been extended by MHEG-3, the new standard MHEG-6 was nevertheless developed. In contrast to MHEG-3, MHEG-6 builds upon existing solutions for data processing. MHEG-6 uses the Java programming language as a basis and defines an interface for the interoperability of MHEG and Java objects.

MHEG-7 -Interoperability and Conformance Testing[Int01a]: This part defines a test suite for interoperability and conformance testing for MHEG-5 engines. Additionally, a format for test cases is defined to allow for detailed and application specific tests..

MHEG-8 - XML Notation for MHEG-5 [Int01b]: Part eight defines an alternative encoding format for MHEG-5 objects based on XML.

In contrast to other standards, MHEG was designed only to be a description language for final-form interactive multimedia presentations. It provides neither a definition of a graphical user interface nor any architectural specification of the execution engine. Although MHEG-1 was not able to gain wide acceptance in the iTV market, its reduced specification, in the form of MHEG-5 is used in several systems. Today MHEG-5 has great industry support and MHEG-5 content (MHEG 5 UK Version 1.06 [Bri03]) is broadcast in the UK and New Zealand. Many extensions and profiles such as Euro MHEG have also been developed and are in use today.

The Multimedia Home Platform (MHP)

The Multimedia Home Platform[1] was specified by the MHP group, a subgroup of DVB. This group was created in 1997 with the goal of developing a standard for a hardware and vendor independent execution environment for digital applications and services in the context of DVB standards. In July 2000 the first version of the MHP standard (MHP 1.0) was published by the European Telecommunications Standards Institute[2] (ETSI) [Eur06c, v1.0.3]. Just one year later, MHP 1.1 became an ETSI standard [Eur06d, v1.1.3]. The latest specification is version 1.2 [Eur07c], in which support for DVB-IPTV was added. It was standardized as TS 102 727 [Eur10c] by the ETSI in 2010. In general, every new version of MHP is an extension of the previous versions.

The MHP specification defines four profiles for different classes of functionalities [Rei04, pages 337-339]. The profiles build upon each other. MHP 1.0 specifies only the first and the second profile. The third profile was introduced with MHP 1.1, and the fourth profile with MHP 1.2. (see also [Mor05, chapter 1]).

1. Enhanced Broadcast Profile (Profile 1): The simplest version of an MHP environment only provides the Enhanced Broadcast Profile. It is aimed for low-cost set-top boxes without a return channel. This profile allows for the development of applications offering local interactivity. Because of the lack of a return channel, applications may only be downloaded from the broadcast stream - the MPEG-2 Transport Stream. Typical applications based on this profile are electronic program guides, news tickers and enhanced teletext applications.

[1] Multimedia Home Platform - http://www.mhp.org/

[2] European Telecommunications Standards Institute (ETSI) - http://www.etsi.org/

2. Interactive Broadcast Profile (Profile 2): In addition to the functions of Profile 1, this profile includes support for a standardized return channel. Through the use of the return channel, an interaction between the audience and the broadcaster becomes possible. This enables the support of applications like televoting, T-commerce or pay-per-view applications. Another advantage of the return channel is that MHP 1.1 applications can also be downloaded via an internet connection.

3. Internet Access Profile (Profile 3): In the Internet Access Profile, Profile 2 is extended to include support for internet applications. This profile just specifies APIs for accessing different internet services and applications, rather than concrete services. By using this profile, typical point-to-point services like email and WWW can be combined with the broadcast world. Online games, chat- and email-applications are often provided using this profile.

4. IPTV Profile (Profile 4): The most enhanced profile is the IPTV Profile. This profile integrates support for DVB-IPTV into the MHP platform. DVB-IPTV is made up of a collection of various specifications for the purpose of delivering DTV using IP. There are various options, such as the broadband content guide (BCG) available for extending the IPTV Profile. BCGs specify signaling and delivery of TV-Anytime [Eur06a] information.

As seen in figure 2.5, the MHP architecture is presented in three layers. These components can be described as follows:

Resource Layer: It represents the different hardware platforms for set-top boxes. Besides

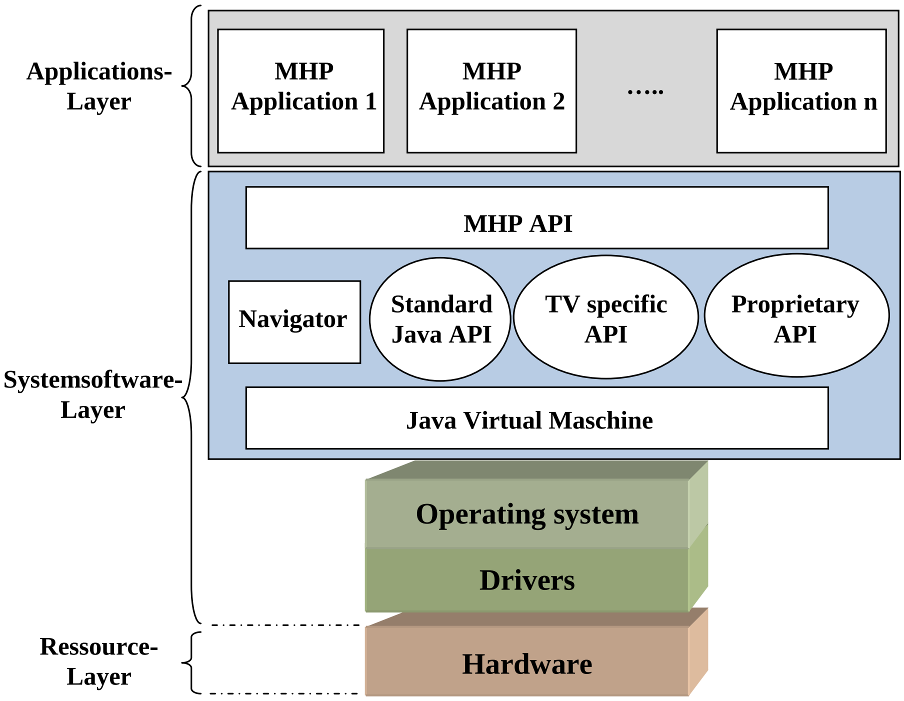


Figure 2.5: The Multimedia Home Platform (MHP) architecture.

different components like CPU, network interface and memory, the TV specific components like the DVB frontend and the MPEG decoder module are also part of this layer.

System Software Layer: Based on the hardware platform, the operating system manages the integration of the hardware and offers basic functions, such as process management, to the MHP middleware. The first component of the middleware is represented by the Java virtual machine. The use of Java offers a hardware-independent common ground for the MHP API. The Java platform of MHP is also known as DVB-Java (DVB-J) [Rup03, pages 199-236]. PersonalJava forms the SUN API part of DVB-J. The main reason for the use of PersonalJava in MHP is the small footprint of PJVMs, which fits perfectly with the limited resources of most set-top boxes. Other important Java components are the Java Media Framework (JMF), the Java TV API and the Home Audio Video Interoperability (HAVi) specification. JMF is used for controlling audio and video content. HAVi (HAVi Level 2 GUI) forms the UI components of MHP. Another important component of the middleware is the Navigator, which is an application manager and offers a user all basic functions for watching TV, e.g. listing and switching channels. The application model and many APIs providing access to DTV-specific functionality of MHP stem from the Java TV specification. MHP applications are commonly called DVB-J applications. The transport protocol's component in the figure consists of all parts and protocols that are necessary for communication via different network interfaces. TV specific protocols such as DVB-SI, DSM-CC and MPEG-2-TS and network protocols such as IP, TCP, UDP and HTTP are used in this component.

Application Layer: As shown in figure 2.5, MHP is able to handle, on top of the MHP API, multiple applications in one JVM. Besides the provided applications, plugins can also be implemented in MHP. These are used to extend the functionality of the platform. There are two categories of applications and plugins in MHP: interoperable (DVB-J-based), and non-interoperable ones. Interoperable ones may be used with all kinds of MHP receivers, on top of the MHP API. The application model of DVB-J applications follows the model of Java TV Xlets. Device specific applications and plugins are not interoperable, because native code or special Java APIs, not available in standard MHP, are used. For that reason, such applications and plugins lose the ability to run with all kinds of MHP receiver.

Besides DVB-J, there is another method for building interoperable MHP applications. The declarative language DVB-HTML, for interactive TV applications has been introduced with MHP version 1.1. Several basic concepts of DVB-HTML such as the application life cycle, had already been introduced with MHP 1.0. The framework of DVB-HTML is based on a selection of XHTML 1.0 modules. For formatting, CSS level 2 is used. ECMAscript and DOM level 2 support form the basis of the dynamic aspects of DVB-HTML applications. The main reason for introducing DVB-HTML was that many companies had expert knowledge in HTML, that they were willing to reuse, and that Java is not the best choice for creating presentation driven applications [Mor05, chapter 15]. DVB-HTML is available for each of the three MHP Profiles.

MHP is already in use worldwide. Among its biggest supporters in Europe are Italy, Finland, Austria and others. Globally, MHP is gaining importance due to the fact that many other iTV specifications relate to MHP (cf. OCAP or ACAP).

Globally Executable Multimedia Home Platform (GEM)

The Globally Executable Multimedia Home Platform (GEM) represents a subset of MHP. GEM 1.0 [Eur05b, v1.0.2], published in 2003, relates to MHP 1.0. GEM 1.1 relates to MHP 1.1, and the specification of GEM 1.2 [Eur07b], published in 2007, relates to MHP 1.2. GEM 1.2 was released as ETSI Standard TS 102 728 [Eur10b] in 2010. The main purpose of GEM is to enable organizations, such as the CableLabs and the ATSC, to define specifications based on MHP with the help of DVB. The goal is to guarantee that applications can be written in a way that will be interoperable across all different GEM-based specifications, platforms and standards. GEM is not a standalone specification, but rather a framework aimed at enabling other organizations to define GEM-based specifications. GEM specifies the APIs and content formats that can be used as a common basis in all interactive television standards. GEM also identifies and lists the components of MHP which are, from a technical or market perspective, specific to DVB. Thus, other organizations are able to use GEM and define their own replacement of the DVB specific components as long as they are functionally equivalent. A specification in which only DVB specific components are replaced is GEM compliant and capable of running MHP applications. Many organizations around the world have adopted GEM as the core of their middleware specifications. Figure 2.6 shows the relationship and the functional replacements between GEM and ARIB, OCAP and ACAP. GEM is referred to in its entirety in specifications like ACAP and OCAP. Differences and extensions have to be defined in detail. Moreover, elements of GEM and concepts originating from MHP are also used in the Blue-ray Disc Java (BD-J) specification to offer interactive applications for Blue-ray Discs.

OpenCable Application Platform (OCAP)

The OpenCable Application Platform[1] (OCAP) is an open middleware standard for interactive TV. The first steps towards this standard were made by a non-profit organization formed by many US cable television system operators called Cable Television Laboratories (CableLabs). The main goal of this initiative was to develop a middleware suitable for most set-top boxes from different vendors and major cable TV system operators. When the work on OCAP began, its European counterpart, DVB-MHP, was already on its way to standardization. For that reason, DVB-MHP was investigated by the CableLabs and many parts where found to be suitable for OCAP. Thus, OCAP is largely based on MHP 1.0. Nevertheless, there are major differences between the distribution standards for DTV in the US and in Europe and, as a result, major differences in the distribution related middleware components. Additionally, some restrictions, made by the Federal Communications Commission[2] (FCC), led to changes and extensions of the middleware

[1] OpenCable Application Platform (OCAP) - http://www.opencable.com/ocap/

[2] Federal Communications Commission (FCC) - http://www.fcc.gov/

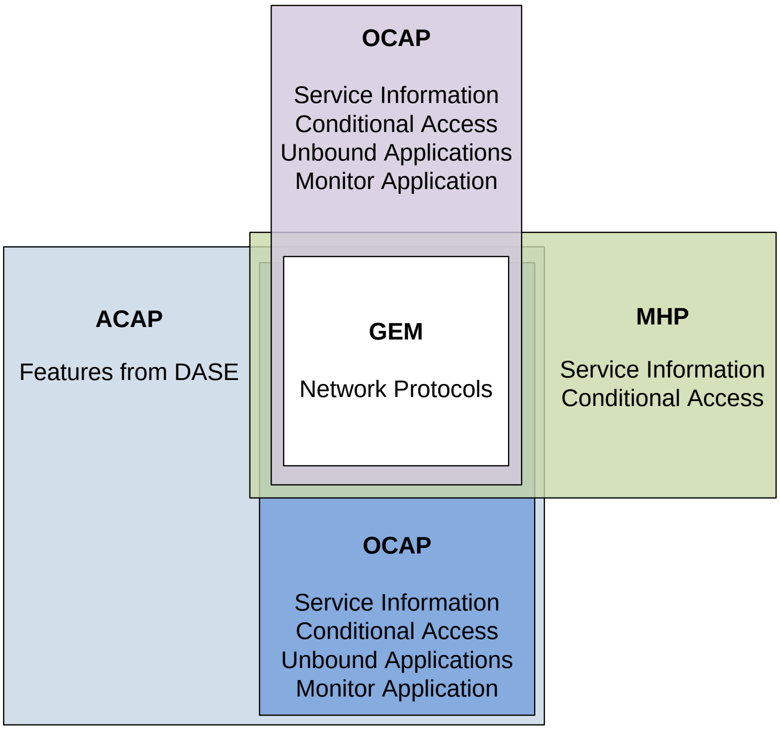


Figure 2.6: Relationships between GEM and ARIB, OCAP and ACAP (cf. [Mor05, figure 18.1]).

components. The OCAP platform defines two profiles, OCAP 1.0 [Cab05] and OCAP 2.0, which are further discussed in the following section.

1. The first profile of OCAP (OCAP 1.0) was published in 2001. OCAP 1.0 defines the basic functionalities of OCAP. Over the years several versions of this profile have been published. In some versions, changes were made on substantial components, which led to the loss of backward compatibility between the versions. The most recent version of this profile is I16, and was released in August 2005.

2. OCAP 2.0 was first published in 2002. This profile expanded upon OCAP 1.0 in several aspects. The most important difference from 1.0 was the inclusion of DVB-HTML support, based on the DVB-HTML extension of MHP 1.1.

In the following paragraphs, the middleware architecture of OCAP will be described. Figure 2.7 provides an overview of the OCAP architecture. The *Hosted Device Hardware* component of the figure represents the hardware platform of an OCAP set-top box which was originally a hybrid analog/digital device. This means, that such set-top boxes are able to support analog as well as digital services. The *Operating System* offers basic services such as task/process scheduling and memory management, and forms a middleware layer between the hardware and the OCAP components. The major functionality of OCAP is provided by the *Execution Engine* and its various modules. Moreover, the engine provides a platform-independent interface built upon the JVM and a set of additional Java APIs.

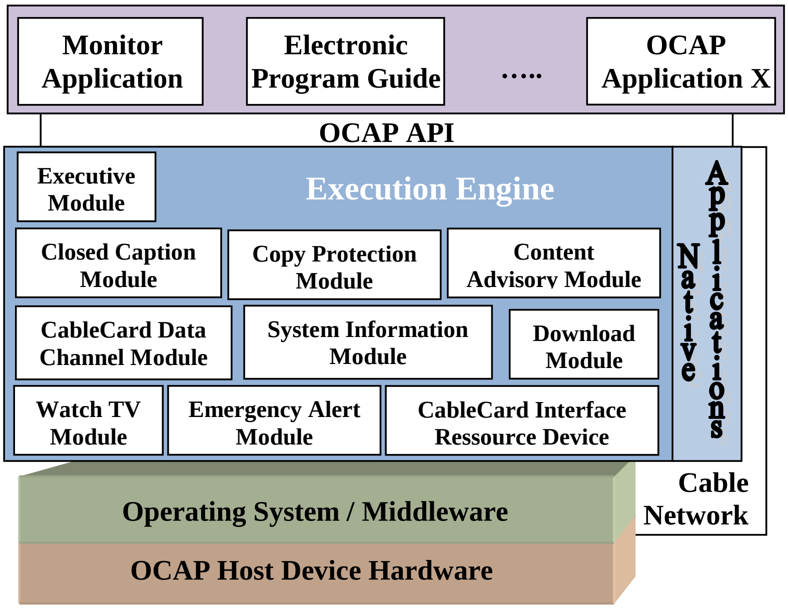


Figure 2.7: The OpenCable Application Platform (OCAP) architecture.

Java support in OCAP, also known as OCAP-J, is based on the DVB-J platform. Taking a closer look at the main modules of the engine, they can be described as follows:

Watch TV Module: This module offers the basic functionalities for watching TV, such as switching channels. It allows the user to watch all unencrypted channels.

Emergency Alert Module: This module is used to broadcast local or national emergency messages. Alert messages provided by the cable network operators force all receivers to show the alert message. This module is, based on the FCC rules for emergency alert system (EAS), a mandatory component of a receiver.

CableCard Interface Resource Device: This module handles all messages for the CableCard hardware that require user-interaction, such as requesting the pin number, and satisfying the communication needs of applications via the module according to the CableCard Interface 2.0 specification.

CableCard Data Channel Module: This module offers baseline functionality for processing data on the CableCard Data channel.

System Information Module: The System Information Module keeps track of service information. After parsing the service information, it offers the information to other modules and OCAP applications. Special types of information, such as emergency alert system messages, are directly forwarded to the appropriate module for further processing.

Download Module: The Download Module keeps track of new updates available for the set-top box. It enables network providers to update their set-top boxes. This is a very important function and represents the only way to get rid of erroneous firmware versions once the set-top boxes are at the customer's home.

Closed Caption Module: The presentation of closed caption text is the main purpose for this module. It is part of the core functions and should work regardless of any extension of the network operator. It is also mandated by the FCC for all analog TV services.

Copy Protection Module: This module controls the copying of analog and digital content. It controls the storage and the output of content according to the Copy Control Information (CCI) delivered by the Conditional Access (CA) system.

Content Advisory Module: The V-Chip functionality, mandated by the FCC, is handled by the Content Advisory Module. The V-Chip provides the option of avoiding the presentation of certain television programs based on their ratings. This makes possible the parental control of the TV consumption of children. The module decodes the V-Chip signal and offers the rating to other modules.

Executive Module: The Executive Module is responsible for launching and controlling applications. The management of stored applications is also handled here. It plays a major role during the start-up phase of the set-top box and loads the initial Monitor application, if one is available. If no application is available, it is responsible for controlling the receiver. While a Monitor application is running, the executive module monitors the life cycle of the Monitor application and re-launches it if it is destroyed.

OCAP applications are in many ways similar to MHP applications. The *Monitor* application, also called Monitor, plays a special role in an OCAP set-top box and represents a unique feature of OCAP. It is either implemented as a single OCAP-J application, or realized by a set of closely related applications. The Monitor application has privileged access to several APIs which are not accessible for ordinary applications. It helps to manage and control the life cycle of OCAP applications, and cares for resource management and security issues. Monitor applications are provided by the cable television system operators and downloaded to the set-top box when it connects to the cable network for the first time. The full functionality of an OCAP set-top box is only available if the Monitor application from the network operator is present. In most cases, only basic functions such as watching TV and using unencrypted services are available without a Monitor. Through the use of this application, a system operator gains much control over the set-top boxes. Further customization can be made by the network operator because of several assumable modules of the Execution Engine, such as the Watch TV Module, the Emergency Alert System Module, and so on. The Monitor application may assume the functionality of these modules and consequently, enable network operators to replace the main functions of the box with his or her own implementations.

As mentioned before, the OCAP standard has a clear focus on cable networks. For that reason, OCAP is widely used by US cable TV system operators. Although OCAP is an open and mature standard, a global use is unlikely because it contains several US-market specific characteristics.

Advanced Common Application Platform (ACAP)

The Advanced Common Application Platform (ACAP) [Adv09] is a middleware specification for iTV applications. It was developed and standardized by the Advanced Television Systems Committee (ATSC), a US non-profit organization dedicated to the development of standards for DTV. The ATSC is made up of members of the television industry, ranging from broadcasters to the semiconductor industry. The ACAP standard was published in 2005. ACAP is primarily based on GEM and DTV Application Software Environment Level 1 (DASE-1) [Adv03] but also makes use of OCAP functionalities. Many sections of the ACAP specification simply refer to parts of these iTV standards. An important capability of ACAP is its ability to support all US DTV systems, cable, satellite and terrestrial television networks. ACAP was the first attempt to harmonize the US iTV market for cable and terrestrial TV.

There are two different types of ACAP applications, procedural applications called ACAP-J and declarative applications called ACAP-X. In general, ACAP-J applications are Java TV Xlets and ACAP-X applications (the ”X” stems from XHTML) are very similar to DVB-HTML applications. ACAP is structured in two profiles. The first profile supports only the first application type, ACAP-J. The second profile adds support for ACAP-X applications. ACAP is based on GEM and DASE. For that reason, the architecture of ACAP and its components are very similar to DASE and parts of GEM/MHP. Nevertheless, there are differences between the broadcast system specific parts and the parts stemming from OCAP.

Hybrid broadcast broadband TV (HbbTV)

Hybrid broadcast broadband TV (HbbTV)[1] is one of the most recent initiatives trying to combine the traditional broadcast domain with services of the broadband respectively the internet domain. One of its main characteristic is that hybrid terminals are able to connect to two networks, commonly a broadcast (DVB) and a broadband (internet) network simultaneously. This enables the use of services, such as TV related digital services and allows access to web portals or video on demand, typically associated with IPTV, on traditional broadcast media such as satellite or terrestrial TV. HbbTV, initially focused on the German and the French markets, has already gained support all over Europe. Developed by a consortium of more than 60 European members, ranging from broadcasters to research institutes and consumer electronic manufacturers, it has been standardized as TS 102 796 [Eur10d] by the ETSI. An interesting fact is that OpenTV Inc, which offers the proprietary iTV middleware solution OpenTV, is also a founding member of HbbTV. Its specification is based on several existing standards and technologies. Among others, the CE-HTML specification [Con07] is used as the application language, the DVB specification TS 102 809 [Eur10a] is used for application signaling and transport via broadcast or HTTP, and parts of the Open IPTV Forum specifications, considering the declarative application environments and media formats[2], have been used. The HbbTV platform makes possible the download and execution of applications defined as a collection

[1] HbbTV – http://www.hbbtv.org/

[2] Open IPTV Forum Release 1 specifications – http://www.oipf.tv/specifications.html

of HTML, CSS, JavaScript, XML and multimedia files. In general, these are the two types of applications addressed:

- Broadcast-independent interactive applications: These applications are downloaded and access all related content via broadband. Typical examples of independent applications are VOD portals, internet TV portals, games and any other kind of web portal or application. A common characteristic is that these applications are not associated with any broadcast service or events.
- Broadcast-related interactive applications: These applications are typically closely related to one or more broadcast services, or at least triggered by a broadcast event. The download of such applications and data access can be done in both ways, broadcast as well as broadband. Typical examples are applications such as an enhanced video text (often referred to as video text 2.0), offering additional information on the current service, and game show applications to participate on votes, etc. These applications may start automatically or can be triggered by user commands.

Note that for privileged operations, such as file system access or triggering recordings, the applications must be signed (trusted). Figure 2.8 shows a rough overview of the main functional components and the HbbTV terminal architecture. The main components of figure 2.8 can be briefly described as follows:

- Broadcast Interface: The broadcast interface is responsible for receiving data from a typical broadcast network, such as via satellite with DVB-S or DVB-S2. Its main aim is to provide A/V content (in HbbTV called linear content) to the processing components. Additionally, application data and stream events are also received and forwarded for processing. Data transfered via broadcast generally uses an DSM-CC object carousel mechanism. Information provided by the Application Information Table (AIT) are used in the *Application Manager* to control the application's life cycle.
- Broadband Interface: Another way to receive application data and (non-linear) A/V content, such as video on demand, is the broadband interface which provides access to the internet. Data received via this interface is forwarded to the *Broadband Processing* step.
- Broadband Processing: All non-linear content, provided as internet data, is handled by the *Broadband Processing* step. A/V content is forwarded to the *Media Player*, whereas applications and data are handed over to the *Runtime Environment* by the *Non-linear content processing* component.
- Broadcast Processing: This type of processing includes steps such as demultiplexing of the DVB stream (cf. section 2.1.2), *Linear content processing*, which includes all functionalities well known from standard DVB terminals, ranging from processing of the A/V content to the extraction of channel and event information (e.g. program descriptions), and AIT filtering for extracting application information. Additionally, the *DSM-CC Client* processes object carousel data and provides it to the *Runtime Environment*.

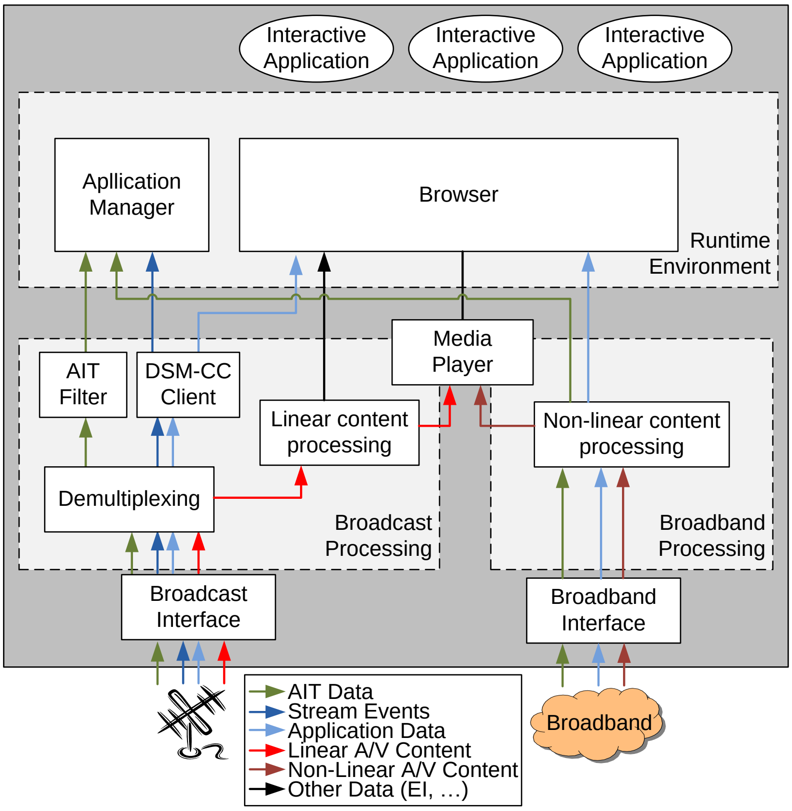


Figure 2.8: The Hybrid broadcast broadband TV (HbbTV) terminal architecture (cf. Figure 2 - ETSI TS 102 796 [Eur10d]).

- Media Player: The playback of linear and non-linear A/V content is controlled and carried out by the *Media Player*. Additionally, it enables media playback in different scaled modes for a better integration of the video into the application's user interfaces.

- Runtime Environment: The runtime environment makes possible the execution and the monitoring of interactive applications. Its main components are the *Application Manager* and the *Browser*. The *Browser* is used for rendering the applications and handling interactions. As applications are build upon typical web standards, such as HTML, CSS and JavaScript, it is very similar to a typical web browser. Directed by the application information of the *AIT Filter*, the *Application Manager* keeps track of the application execution. It is responsible for starting and stopping as well as monitoring each step of the applications' life cycles.

HbbTV distinguishes between different presentation modes with a different balance between video and interactive application. Among those available, the visual prompt just displays a small visualization (prompt) for the application in the video. Others are the overlay mode, in which the video is integrated into the application's presentation, and the information-only mode, where no video is presented. In HbbTV, no user input device is specified, however the use of a conventional remote control, or at least a similar concept of pressing buttons on a remote control is recommended.

YouView

Project YouView[1] (formally known as project Canvas) aims to develop an open standard platform for IPTV in the UK. First announced on December 2008, it is currently a partnership between free-to-air broadcasters such as the BBC, Channel 4, Five and ITV, and broadband network operators like BT, Arqiva and Talk Talk. It is focused on broadband set-top boxes and will provide seamless access to a wide variety of applications and web content. YouView will extend traditional TV with the features of catch-up TV, where TV content from the last seven days may be accessed, and in addition on demand content can be accessed. As an open platform based on embedded Linux it can be easily enhanced by new applications. The technical specifications of the project have been developed in cooperation with several CE device manufacturers. In YouView the UK Profile of MHEG-5 (cf. MHEG paragraph above) is again supported. In order to make the creation of applications for content providers easier, the additional support of a W3C profile (HTML, CSS and JavaScript) and a Flash profile by the YouView presentation engine is planned. First YouView devices are scheduled to become available in the first half of 2011.

Other Platforms

Aside from the open standards presented, a wide variety of proprietary solutions for iTV middleware exist. One widely used system is OpenTV[2]. Other products are Mediaroom[3] from Microsoft and MediaHighway[4] from NDS. Since these platforms are proprietary, a further description is not within the realm of this thesis.

2.1.4 The History and Future of iTV Standards

In their latest or most advanced versions, all of the standards presented feature comparable capabilities. The main differences lie in the initial objectives of the specifications. These goals included the development of representation languages, middleware specifications and ”all-around” standards covering all parts of the iTV domain. It should also be mentioned that regional distinctions led to differences and additional components (cf. OCAP, ACAP, etc.). Nevertheless, most of the standards are strongly interrelated, as shown in figure 2.9. In considering the way iTV applications are constructed, it becomes apparent that Java and different web standards such as HTML, CSS and JavaScript are fundamental building-block technologies. Taking a closer look at the evolution of iTV standards, it is easy to see that Java was the leading technology in the beginning of the iTV standardization process. The application model of many iTV applications is Java-based. For this reason, it can be said that a clear relationship between Java and most iTV standards is present. In figure 2.9 this relationship is indicated by arrows from Java to MHEG, DAVIC and Java TV and

[1] YouView – http://www.youview.com/

[2] OpenTV - http://www.opentv.com/

[3] MS Mediaroom - http://www.microsoft.com/mediaroom/

[4] MediaHighway - http://www.nds.com/solutions/mediahighway.php

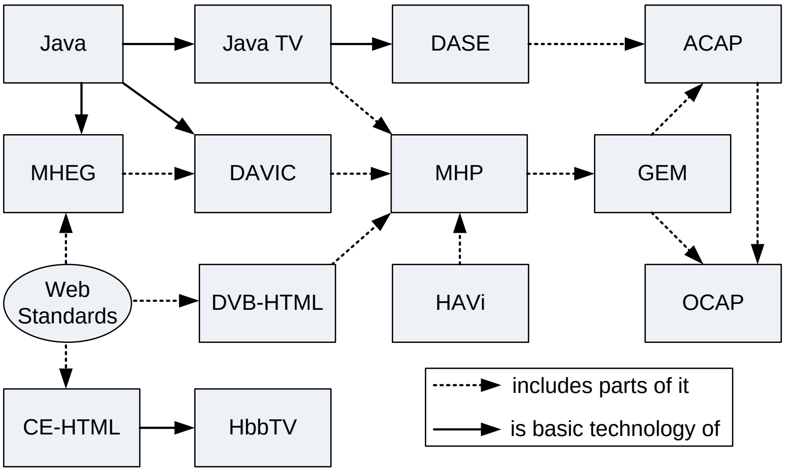


Figure 2.9: Relationships between the presented interactive TV standards.

the connections of these standards to all other standards. Web standards were included as a kind of alternative technology or at least as an extension of building applications in later developments of most standards. A fundamental specification for this integration is DVB-HTML which adds support for technologies such as XHTML, CSS, DOM and ECMAScript. The first open standard for iTV was MHEG, initially conceived of as a description language for multimedia presentations. A scripting language was, however, soon included in order to allow the implementation of advanced applications. In spite of other versions of the MHEG specifications, MHEG-5 and its extension MHEG-6 are most relevant. These were partly included in the DAVIC specifications, which were the industry standard for interactive digital audio-visual applications and broadcast. Like DAVIC, MHEG-6 also makes use of the Java programming language to maximize interoperability. The Java TV API was developed to provide a pure Java environment to control and run applications on TV receivers, such as set-top boxes.

Although figure 2.9 does not indicate a relationship between DAVIC and Java TV, several concepts from DAVIC (e.g. controls defined in DAVIC) are used. MHP is a comprehensive specification for iTV middleware platforms. It includes elements and concepts of DAVIC, Java TV and several parts of HAVi (e.g. UI components). Furthermore, a reciprocal knowledge transfer also took place during the standardization of MHP and Java TV. The OCAP standard was introduced specifically for cable networks in the US. OCAP specifies a middleware platform with several cable network and FCC specific components. The OCAP standard reuses major sections of the MHP specification. Since many standards reference MHP, a subset - GEM - has been defined. GEM forms a common core for many MHP related standards. As indicated in figure 2.9, GEM is fully included in new versions of the OCAP specification and ACAP. ACAP is a young standard which supports all common DTV systems in the US. Because ACAP was developed by the ATSC, the former ATSC standard DASE was included in ACAP.

In contrast to older standards, most recent standards, such as HbbTV, rely solely on web standards for their applications and do not further integrate Java as a base technology into the specifications. Note, that the support of web standards in HbbTV is based on CE-HTML, which defines a much smaller subset for web standard support compared to comprehensive support in DVB-HTML. Thus, the role of Java in the future of iTV is also controversial, although it has been chosen as application technology BD-J for the Blue-ray standard. The emergence of the iTV standards, especially GEM, shows the trend to a harmonization of the iTV market. HbbTV and YouView seem to go their own way in the iTV domain, although many ideas and solutions are similar to proprietary platforms and other iTV standards. Several concepts from the "HbbTV world," have already been introduced earlier by MHP and GEM.

2.2 TV Metadata and Standards

TV metadata and standards are among the most important elements making it possible for viewer to navigate through a constantly growing number of TV offers. These have become even more important with the introduction of digital TV. Aside from the needs of the viewers, the needs of other players in the TV value chain also need to be addressed. This diverse group includes the content creator, the broadcaster, the advertiser, etc.

TV metadata is created at all stages of the TV value chain. It contains information such as movie scripts, the parameters of the broadcast network, the coding of the content, TV channels, TV programs and other metadata of stages ranging from the pre-production (design and conceptualization of content) over the production (content creation), post-production (editing, combination of media items) and the delivery (broadcast, streaming) to the presentation (content consumption) of the content. In the case of interactive Television (cf. section 2.1), the program takes its final shape in the presentation step, whereas "traditional" TV takes its in the post-production step. Thus, most standards are designed for a certain stage. Figure 2.10 shows an overview of the audiovisual media production process. Metadata in the broadcast area can be categorized into the following three main groups:

- Technical metadata, such as network information, as well as capture and format information.
- Administrative metadata, which makes possible the management and administration of multimedia data such as rights, project management, creation and reproduction information.
- Descriptive metadata, which describes the content of the object, such as human readable descriptions or related information (e.g. title, actors, synopsis).

In this work, we concentrate on descriptive metadata used in the delivery and presentation stage of this process. TV metadata about programs, also called Electronic Program Guide (EPG) data, are at the foundation of our metadata oriented personalization concept as described in chapter 6. In our system, the amount of elements, the availability and the quality of the program metadata are crucial for providing adequate program recommendations. Program descriptions commonly include basic elements such as title, subtitle, a short synopsis, start time, duration and channel. Often, more advanced elements such as producer, actors, moderator, language, year and country or information for classifying programs like genre, parental guidance rating or program rating are also available. Nevertheless, the presence of specific metadata elements strongly depends on

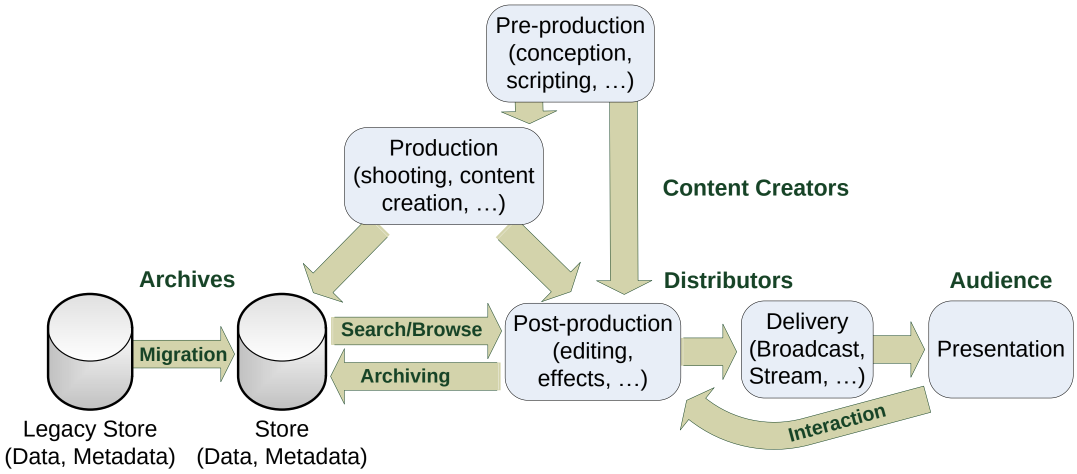

Pre-production (conception, scripting, …)
Production (shooting, content creation, …)
Content Creators
Archives
Distributors
Audience
Migration
Search/Browse
Archiving
Post-production (editing, effects, …)
Delivery (Broadcast, Stream, …)
Presentation
Interaction
Legacy Store (Data, Metadata)
Store (Data, Metadata)

Figure 2.10: Overview of the audiovisual media production process (see figure 1 on page 67 in [Lux08]).

the chosen standard and metadata source in use. In general, it is important to differentiate between the formation and presentation of TV metadata and the metadata source. In the first part of this section, we will discuss different sources for EPG data. Accordingly, sections 2.2.1 to 2.2.6 focus on the introduction of several available metadata standards and also discuss the role of proprietary formats. We will focus on the most significant ones, which are the following:

- Section 2.2.1 – Digital Video Broadcasting - Service Information (DVB-SI), a component of the European standard for Digital Video Broadcasting
- Section 2.2.2 – MPEG-7, which focuses on the content description of multimedia in general and as a result, provides a comprehensive framework for handling and describing all forms of multimedia content
- Section 2.2.3 – TV-Anytime, which addresses the needs of all players in the TV realm, including content providers, viewers as well as advertisers
- Section 2.2.4 – XMLTV, a small and flexible quasi standard in the world of program metadata
- Section 2.2.5 – BBC Program Ontology, a very recent development of a web ontology covering program data
- Section 2.2.6 – Proprietary formats, which are used by different program data providers and electronic program guides

Other important existing standards are P_Meta [Eur07a] of the European Broadcasting Union (EBU), which is focused on the exchange of program-related metadata in the business-to-business area, EBU Core [Eur05a], which defines a minimum set of information needed to describe radio and television content. Still further examples are BBC Standard Media Exchange Framework (SMEF) [Cor00], which is designed as an internal description format for all kinds of media items of the BBC, and the Material Exchange Format (MXF) [Soc04], which is an file format specification for wrapping metadata in the production process. These are not discussed in this chapter mainly due to their negligible relevance in the TV consumer domain.

Finally, section 2.2.8 will conclude this section with a short comparison of the different metadata standards and sources.

2.2.1 DVB-SI

Digitalization of television networks within Europe is based on the Digital Video Broadcasting (DVB) series of standards. Defined by ETSI (ETSI EN 300 468 [Eur08]), the Service Information (SI) Standard lays out the procedure for the processing of additional data within DVB. Generally, the SI specification can be seen as a kind of metadata specification, as well as a transport mechanism within DVB. Figure 2.11 shows the structure of DVB SI, complete with its different tables. Each table holds a specific category of metadata, ranging from technical information to EPG data. In the following section, we will briefly discuss the different tables.

- PAT: The Program Association Table (obligatory for the current TS) holds a list of all programs in the current multiplex of the transport stream (TS) (cf. section 2.1.2). By including the identifiers of the Program Map Tables, all associated information for a program can be found.

- PMT: The Program Map Table references all packets concerning a program.

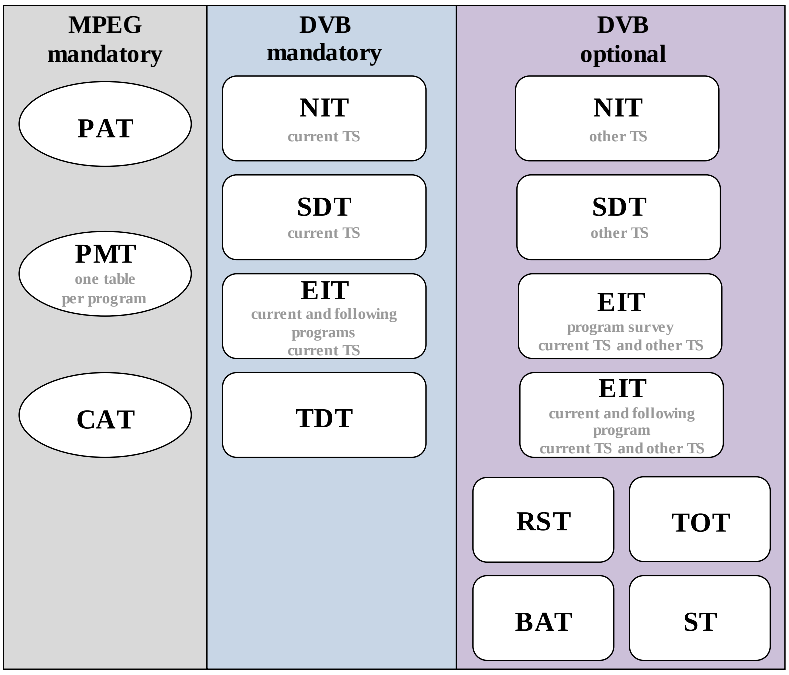


Figure 2.11: Overview of the Program Specific Information (PSI) and Service Information (SI) tables (see figure 5.11 in [Rei08a]).

- CAT: The Conditional Access Table contains "private data" for conditional access (e.g. data used by pay TV providers).
- NIT: The Network Information Table contains important information about the network (e.g. the transponder number).
- BAT and SDT: The Bouquet Association Table and the Service Description Table provide information about available services, the service provider and about the organization of services in bouquets.
- TDT and TOT: The Time and Date Table and the Time Offset Table contain information used in synchronizing the clock (e.g. a current timestamp and the offset to the local time).
- ST: The Stuffing Table is used for marking data and sections as invalid.
- EIT: The Event Information Table forms the basis for the Electronic Program Guide. Each program has its own EIT. The mandatory "present and following event table" contains at least the title, start time and duration of the currently shown and of the upcoming program (also called event). Genre, category, a short description as well as aspect ratio, audio format and parental rating may also be listed. DVB-SI recommends the use of a hierarchical 2-level classification of programs. The top level of the hierarchy consists of ten broad program categories such as movie/drama, news/current affairs, sports and children/youth program. On the second level, there are from 6 to 16 sub-categories per top level. For instance, for movie/drama, sub-categories such as comedy, romance and detective/thriller are defined. Nevertheless, this categorization is often said to be too rough for a proper content classification (e.g. no identifier for golf or car racing exist). Optional schedule event tables can contain program/event information for up to 64 days in advance.

For a detailed discussion of the different DVB tables, the reader is referred to [Rei08a, Eur08]

As most of the programs' elements are not mandatory, there is no way to assure that consistent and homogeneous program data is available in the TS. Moreover, the availability of information for programs beyond the present and the next event can also not be guaranteed.

2.2.2 MPEG-7

The ISO/IEC 15938 [Int02] standard MPEG-7 was passed in 2002 by the Moving Picture Experts Group. It focuses on multimedia content description and offers tools for describing content of different forms (e.g. images, audio, video, speech), different formats, at different granularity (e.g. frame, shot, sequence) and with different aims (e.g. content management, search, filtering). MPEG-7 documents, which are coded in XML, must comply with the MPEG-7 XML Schema definition. For effective storage, the standard offers a ”Binary Format for MPEG-7 Description Streams” (BIM). Formally known as ”Multimedia Content Description Interface,” MPEG-7 offers a comprehensive framework for describing and managing media content. The standard is organized into 12 parts, which will be briefly introduced in the following section.

MPEG-7 Systems

This component mainly covers the delivery, representation (textual, XML based, or binary, BiM based) and synchronization of descriptions and their content. It introduces the tools needed for the management and usage of multimedia descriptions. Additionally, a MPEG-7 reference decoder is specified in this part.

MPEG-7 Description Definition Language

The Description Definition Language (DDL), based on an XML Schema, is one of the main parts of MPEG-7. It facilitates the specification of description schemes and descriptors, and as a result, provides a language for defining the structure and content of multimedia descriptions. A descriptor (D) specifies the representation of multimedia content's features, ranging from low-level, audio or visual features (e.g. dominant color of an image), to high-level, semantic descriptions (e.g. a person in a video). Description schemes (DS) combine individual descriptors and other schemes, and define the structure and semantics of the relationships between its components. The DDL enables users to define domain or application-specific descriptors or description schemes. Figure 2.12 shows an overview of these main elements and their interactions.

MPEG-7 Visual

MPEG-7 Visual provides a broad collection of basic descriptors for the visual domain (images and videos). Based on low-level features such as color (e.g. dominant color descriptor, color structure descriptor), texture (e.g texture browsing descriptor, edge histogram descriptor), shape (e.g. contour-based shape descriptor, 3-D shape descriptor) and motion (e.g. camera motion descriptor), detailed descriptions for generic visual content can be built. These descriptors and their combinations are applicable for identifying, comparing or searching for images and videos. Aside from these basic descriptors, application-specific descriptors, such as a descriptor for recognizing faces, can also be formulated.

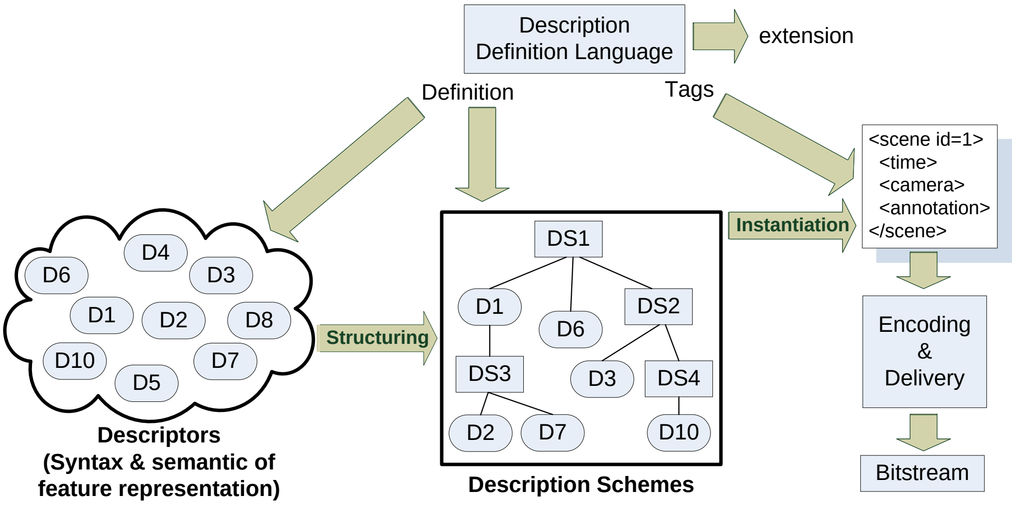


Figure 2.12: Overview of the MPEG-7 main elements (see figure 2 in [Mar10]).

MPEG-7 Audio

MPEG-7 offers a basic group of low-level audio descriptors, facilitating the development of different applications in the audio realm, such as audio search engines. The Audio part offers seventeen temporal and spectral descriptors (e.g. audio power descriptor, audio signature descriptor) making possible usage scenarios, such as query by humming or retrieval by melody description.

MPEG-7 Multimedia Description Schemes

The MDS specify generic multimedia descriptors and description schemes. These range from basic common descriptors (e.g. basic data types, the type hierarchy) to complex specialized description tools (e.g. for content management). Such elements are grouped into the categories Basic Elements, Content Description, Content Management, Content Organization, Navigation and Access, and User Interaction (cf. figure 2.13).

- Basic Elements: Most fundamental elements such as basic data types, constructs to link the different MPEG-7 elements (D and DS) together, and tools for the organization and packaging of descriptions are grouped under this heading.
- Content Description and Content Management: These categories offer adequate descriptor schemes to represent detailed information, usage information, structural and conceptual characteristics of a given piece of multimedia. Program information including title, creators, genre, creation location, guidance information and descriptions on a semantic level, can be modeled based on several descriptors. More detailed content descriptions, including segment and perceptual feature descriptions, are also covered.
- Navigation & Access: This category offers DS for hierarchical and sequential summaries, multiple view support and different variations in scale, coding and modality.

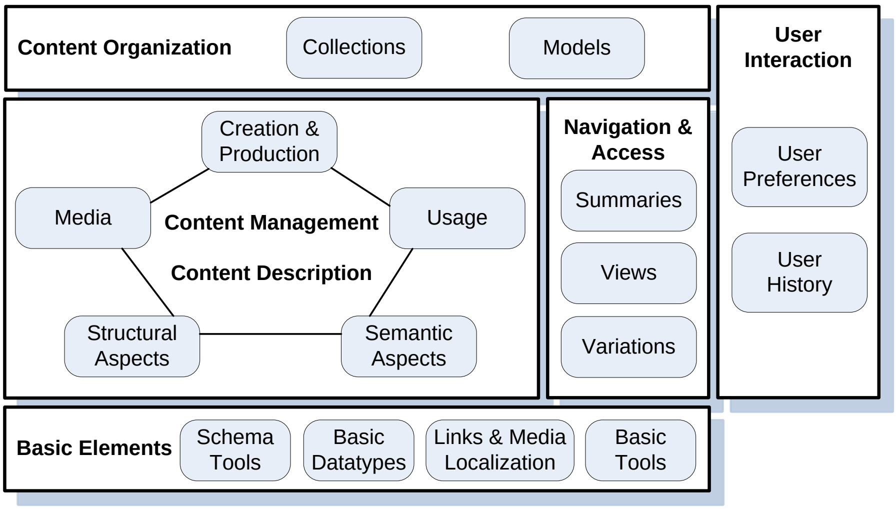


Figure 2.13: Organization of the MPEG-7 Multimedia Description Schemes (see figure 4 in [Mar10]).

The main goal of these DS is to simplify the way users navigate through and access multimedia content.

- User Interaction: Based on the UserInteraction DS, the preferences of users (UserPreferences DS) and the usage history (UserHistory DS) of a user can be modeled. The UserPreferences DS can be used to represent the user's explicit profile containing his or her Filtering-, Search-, Browsing-, Recording- and Creation-Preferences. Additionally, importance values for each preference, privacy characteristics and preference condition, including time and place, can be specified. The user's interaction with the system can be recorded in varying levels of detail, using the UserHistory DS. His or her explicit defaults are represented with the aid of the UserPreference Description Scheme. This description scheme contains context related elements as well as elements related to the viewing environment.
- Content Organization: The Content Organization provides DS to facilitate the description and organization of collections made up of audio-visual content, or spatial or temporal content segments. Based on CollectionStructure DS the relation between different collections can also be modeled.

MPEG-7 Reference Software

This part provides reference software for all important elements of the standard and for conformance testing of MPEG-7 elements (D, DS, DDL). Additionally, tools for extracting Descriptors and several experimental tools are included.

MPEG-7 Conformance Testing

This element encompasses all steps of the process concerned with conformance testing and provides appropriate testing guidelines.

MPEG-7 Extraction and Use of MPEG-7 Descriptions

In order to enable proper use and extraction of MPEG-7 Descriptions, this part offers guidelines and examples for these steps.

MPEG-7 Profiles and Levels

MPEG-7 makes it possible for a user, within a profile, to define an appropriate subsets of the MPEG-7 standard for a specific area of application. Additionally, these profiles can be organized on different levels that define sets of constraints in order to reduce the complexity. In general, each instance document which is compliant with a specific profile, must also be compliant with the whole standard. The reverse, however, does not hold.

MPEG-7 Schema Definition

This part provides information on the extraction of different descriptors and on the usage of several tools included in the reference software of the standard.

MPEG-7 Profile Schema

This part provides different schema definitions for various levels of profiles, ranging from the Simple Metadata Profile (SMP) to the User Description Profile (UDP) and the Core Description Profile (CDP). The SMP facilitates the simple tagging of multimedia. UDP makes possible the description of user preferences and usage patterns. Finally, the CDP facilitates the creation of detailed descriptions of multimedia content and collections.

MPEG-7 Query Format

The MPEG-7 Query Format (MPQF) provides a standardized interface for querying multimedia content in a retrieval system independent way. It provides a format for the query (input), the retrieval result (output) and for selecting, querying and managing of multiple retrieval systems and services (management).

For a detailed description of each specific part the interested reader is referred to [Mar10, Kos03, Sal02].

2.2.3 TV-Anytime

TV-Anytime (TVA), published as ETSI Standard TS 102 822 (Part 1 to 9), is a comprehensive specification that addresses the needs of all groups involved in TV as producers, content providers, consumers and advertisers. TVA is an open and platform-independent specification developed by the TV-Anytime Forum[1]. It is based on detailed metadata about TV content. TVA facilitates the search, choice, acquisition, handling and access of media across different sources and aims (e.g. application in Personal Digital Recorders). TV-Anytime includes definitions concerning the areas of metadata (cf. Part 3) and of content referencing (cf. Part 4). The TVA specifications are organized in two phases. TVA phase one mainly concentrates on the definition and support of metadata authoring, feature-based searching for interesting programs, accessing TV schedules, adding segmentation information to recorded content and aggregating metadata, focused on unidirectional networks. Phase two focuses on delivering content, targeting consumers based on their profiles (e.g. target advertising), distributing and redistributing content (e.g. content sharing) and including additional content types (e.g. web sites, music, games, enhanced TV). Phase two can be seen as the future of digital entertainment, beyond traditional TV (cf. section 2.1). TVA makes use of several parts of MPEG-7. It uses the MPEG-7 DDL to describe content metadata and BIM as an efficient binary XML encoding mechanism. Moreover, subsets of the MPEG-7 DS and MDS are also referenced and included in TVA. TVA is organized in 9 parts, which will be briefly introduced below.

Part 1 and Part 2: Benchmark Features and System Description

These two parts introduce, on an informative level, the main structure and features of TVA phase one and phase two. For both phases, the key benchmark business models describing targeted use cases on an abstract level are described during these parts. A

[1] TV-Anytime Forum – http://www.tv-anytime.org/

high-level system architecture based on a TVA system is presented, providing a channel for interaction and user responses. Using this scenario, the main players in the value chain, namely the service provider, the transport provider, the consumer and their relationship to one another are described.

Part 3: Metadata

Part 3 addresses the TVA Metadata definitions and structure in detail. It is organized into 4 subsections ”Phase 1 - Metadata schemas,” ”System aspects in a uni-directional environment,” ”Phase 2 - Extended Metadata Schema” and ”Phase 2 - Interstitial metadata.” Descriptive information about content, such as typical program descriptions, are presented here. These descriptions are targeted at consumers or agents in order to attract content and facilitate the selection of content in the acquisition step. As a result, this information is called the ”attractor.” TV-Anytime defines the following types of metadata:

- **Content Description Metadata** covers invariable metadata element that are independent of any specific instantiation. Among others, elements like title, genre, actors, summaries and reviews can be used to describe the content. In TVA, a piece of content can be either a ”program” described as an editorial coherent piece of content, a ”program group” representing a description for a whole group, such as a series, or a ”program location,” which describes the relationship between a program and its concert instantiation (e.g. in a program schedule). Multiple ”Program location” descriptors can be grouped to form a program schedule.

- **Instance Description Metadata** is used to describe meaningful differences between instances of the same content. It contains variable data such as location, usage conditions and the content’s audio/video format.

- **Consumer Metadata** is made up of user preferences and usage history based upon the MPEG-7 UserPreferences DS and UserHistory DS. These definitions ensure the interoperability and exchangeability of user and usage information among different services, applications and all groups involved in TV. The usage history commonly contains a list of user actions, such as play or switch channels. By tracking and monitoring the content viewing habits of individual users, applications like personalized TV guides are made possible. Moreover, such information may also be used for target or group oriented marketing. By using the UserPreferences DS, systems are made possible to specify user preferences for various activities, including searching, browsing, filtering, selecting and consuming content in different usage contexts.

- **Segmentation Metadata** facilitates the definition, access, manipulation and description of temporal content segments, such as shots or scenes. Additionally, the organization of such segments into segment groups is possible.

- **Metadata Origination Information Metadata** contain detailed information about the provider of metadata, including copyright notices.

- **Interstitial and Targeting Metadata** defines both metadata and a framework for controlling playback and the interstitial replacement of pieces of content at playback time, based on the characteristics of a consumer's environment and context.

Figure 2.14 shows an overview of all TVA metadata types and the related tables for providing data.

Part 4: Phase 1 - Content Referencing

TVA defines a mechanism, known as the Content Reference Identifier (CRID), to uniquely reference a piece of content. Aside from the identification function, it is also used as a link to connect different content-related descriptions, such as multiple instance descriptions and the content description. Thus, it represents the main element used to structure and connect content and metadata. Each program has its own CRID, whereas repeated content generally is assigned the same CRID. A CRID has the following structure:

$$CRID : // < authorityname > / < locator >$$

In CRIDs, the authority name is resolved to its DNS name. These identifiers are issued by an authority, e.g. a broadcaster, that resolves the CRID to a location. Related objects, such as programs, groups of programs (e.g. series) and events (e.g. program start), are linked together using CRIDs. When referencing a program with a CRID, it resolves to the program's channel and starting time. However, in the case of referencing a group, it resolves to a list of CRIDs of the individual programs of that group.

Part 5: Rights Management and Protection (RMP)

This part is used for the expression and enforcement of the rights of holders and defines content usage conditions. RMP is targeted on baseline RMP information, security tools, APIs (service API and program API) and device interfaces. It is not restricted to the broadcast domain and therefore also applies in post broadcast domains. Especially in the case of content sharing, such usage rules are indispensable. This part is subdivided into ”Sub-part 1: Information for Broadcast Applications” and ”Sub-part 2: RMPI binding.”

Part 6: Delivery of Metadata over a Bi-directional Network

Part 6 describes the exchange of metadata between TVA devices over a bi-directional network. Furthermore, it also covers the discovery of metadata services, based on web service technologies, and the definition of user profile services for retrieving and updating user-centric information. This part is composed of three sub-parts ”Service and transport,” ”Phase 1 - Service discovery” and ”Phase 2 - Exchange of Personal Profile.”

Part 7: Bi-directional Metadata Delivery Protection

This part covers, in addition to RMP, new protection issues introduced by bi-directional metadata delivery. Among others, message integrity, authentication of service providers and encryption are described in this part.

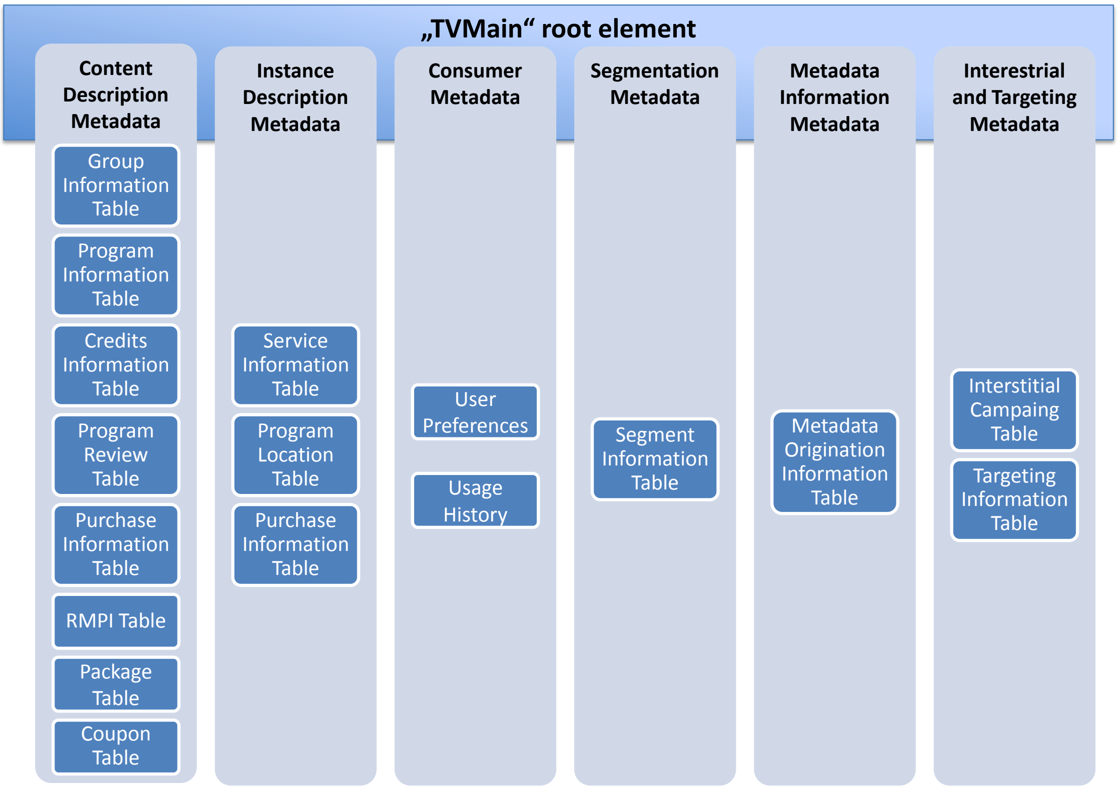

„TVMain“ root element
Content Description Metadata
Group Information Table
Program Information Table
Credits Information Table
Program Review Table
Purchase Information Table
RMPI Table
Package Table
Coupon Table
Instance Description Metadata
Service Information Table
Program Location Table
Purchase Information Table
Consumer Metadata
User Preferences
Usage History
Segmentation Metadata
Segment Information Table
Metadata Information Metadata
Metadata Origination Information Table
Interestrial and Targeting Metadata
Interstitial Campaing Table
Targeting Information Table

Figure 2.14: TVA Metadata types and their organization (see figure 7 in [Eur06a]).

Part 8: Phase 2 - Interchange Data Format

This part defines a delivery network agnostic format for program metadata and content referencing. It facilitates the use and transfer of metadata coming from different sources and in different formats (e.g. a program description from a website) to a TVA device. Additionally, the adaptation of non-TVA services for providing data to TVA clients is covered in this part.

Part 9: Phase 2 - Remote Programming

In part 9, the main focus is on the remote control of a Personal Digital Recorder (PDR) from different devices and locations (e.g. setting a recoding timer remotely from a mobile phone). As an extension to the PDR, the concept of the Network Digital Recorder (NDR) and its management is also introduced in this part.

2.2.4 XMLTV

The non-proprietary framework XMLTV[1] contains a comprehensive XML-based metadata format and a collection of tools (most are Perl based) to extract program information from various sources. Maintained by the XMLTV project, it facilitates the search for and the

[1] XMLTV – http://www.xmltv.org/

manipulation of program information. XMLTV separates front-end and back-end. The front-end mainly focuses on the presentation of data to the user, providing high levels of usability by offering filtering and search functions. The back-end is used to obtain the program data by using different metadata grabbers, such as program metadata extraction tools for different broadcaster websites. The XML syntax, as defined in the xmltv.dtd, represents a common foundation of both sides. XMLTV is commonly used and supported by media centers and personal video recorders, such as the open source projects Freevo[1], MythTV[2] and MeediOS[3], as well as the commercial products Sage TV[4] and Beyond TV[5].

In contrast to most other metadata standards, the XMLTV format is structured and organized in a way similar to how consumers use TV Guides. It includes all channels combined into one listing. The format defines the following three main elements:

- **TV** represents the XML root element of the format. It describes general information about the metadata provider (called ”source”) including the URL or name, and about the generator used to create the TV listing.
- **Channel** defines a TV channel with a unique ID, its display names (e.g. in different languages), an associated icon and url. Generally, the channels included in a program listing are stated first, followed by the ”program” elements.
- **Program** describes a particular event in the program schedule. Each ”program” has an associated channel, as referred to by its ID, as well as a start element. Among other types of information, the title, sub-title, description, category, credits (actors, directors, guests, etc.) and star-ratings can be described. On a more technical level, each program may provide information about audio or video, such as aspect ratio or quality (SDTV or HDTV).

XMLTV is widely used in the US where TV listings are freely available from a feed provided by Tribune Media Services. Program information is commonly extracted from the web pages of local data providers and broadcasters. In spite of the legal US listings, grabbing data from web pages is, in most cases, not granted by the data supplier’s terms and conditions. This introduces legal issues as well as technical issues when the web site’s layout is changed. In Germany, the collecting society VG Media charges considerable fees for EPG data from most German private broadcasters. Thus, there is no source of freely available data to allow for further processing.

2.2.5 BBC Programmes Ontology

In 2007 the BBC initiated a new project to develop a semantic web ontology covering program data called ”The Programmes Ontology.”[6] The ontology offers web identifier for different concepts, such as brands, series and episodes. It is organized into the following four main metadata groups:

[1] Freevo – http://freevo.sourceforge.net/
[2] MythTV – http://www.mythtv.org/
[3] MeediOS – http://www.meedios.com/
[4] Sage TV – http://www.sagetv.com/
[5] Beyond TV – http://www.snapstream.com/products/beyondtv/
[6] BBC Programmes Ontology – http://www.bbc.co.uk/ontologies/programmes/

- **Content** coves categorical information about programs and the relationships between these categories (e.g. a series and the episodes in it). Different properties, such as genre, synopsis, author and director make detailed program descriptions possible.
- **Medium:** offers concepts needed for the description of a concrete broadcasting service (such as BBC News) with its associated channel and broadcaster.
- **Publishing** links the episodes with their broadcast on a service. Because episodes may have different versions (e.g. different audio or with sign language), a broadcast connects a service with a particular version.
- **Temporal Annotations** offer annotations related to a version's segment (e.g. subtitles or description of an episode version's part).

Based on these definitions the exposition, interchange and interlinking of schedule and program information should be facilitated. Figure 2.15 shows a rough overview of the BBC Programmes Ontology's main concepts and their interaction.

2.2.6 Proprietary EPGs

Many metadata providers use their own specialized and proprietary formats to represent, process and deliver data. Usually this is done because knowledge is missing about existing standards or technological barriers, and additional effort related to the application of such standards. The online presence of TV stations is also primarily focused on the layout and styling of their program schedules, rather than on a "proper" data format. For instance, epgData.com[1] from the Axel Springer AG provides EPG data in a proprietary and poorly

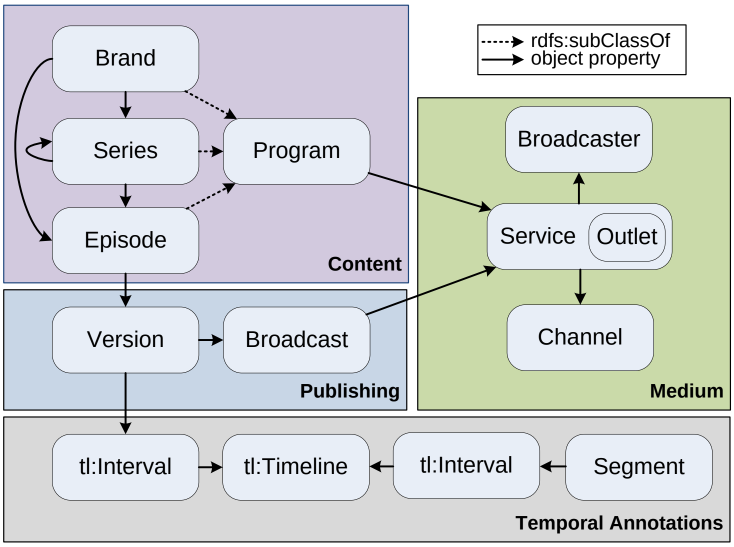


Figure 2.15: Excerpt of the BBC Programmes Ontology (source: see footnote 6 on page 41).

[1] epgData.com – http://www.epgdata.com

structured XML and in a plain text format. Generally, the use of different proprietary formats significantly hurts interoperability and makes the combined use of different data providers hardly possible. Furthermore, despite of well defined standard formats, the specifications and documentation of proprietary formats is often inadequate. As a result, the semantics of different metadata elements and values can also be seen as controversial.

2.2.7 Metadata Sources

Program metadata is collected from a variety of different data providers and sources. In the following we focus on those offering EPG data in German. EPG data is commonly provided by the TV stations (content provider) or external suppliers. As program information from different EPG suppliers often varies considerably concerning the following aspects:

- Channel coverage: How many of the channels, available on the consumer side, are covered by the source of information?
- Element Coverage: To what extent are metadata elements available and used to provide sophisticated and precise program descriptions?
- Availability and Accessibility: Is the information available and can it be easily accessed at the consumer side?
- Consistency: Are descriptions provided in a uniform manner concerning the used parameter values (e.g. genres) and metadata elements (e.g. the same program on different channels and times should be at least identifiable)?
- Up-to-dateness: Are the program descriptions up-to-date?

Aside from the analog way, in form of printed TV guides, there are two other main distribution channels for EPG data, the broadcast itself and external sources accessed via the internet.

Most TV standards enable the transport of different kinds of data, simultaneous to the actual video and audio content of TV programs. As described in section 2.2.1, in DVB this data is included in the so called Event Information Table (EIT). EIT program information are directly provided by the TV stations, typically describing the upcoming programs for the next few days. Thus, up-to-dateness of information and full channel coverage can be expected. A common problem with EIT program information is, that only a small subset of the defined metadata elements are mandatory. Thus, TV channels provide strongly varying program descriptions concerning the element coverage. In some cases, the program metadata available via DVB only include mandatory information on the current and upcoming programs (start time, title and duration). Most description elements such as the category, genre or actors, which are important for TV recommendation generation, are missing. Furthermore, consistency between different TV channels cannot be guaranteed with any level of confidence. However, because the data is provided by the broadcaster, it can be counted on to be both available and reliable.

On the internet, a wide variety of different EPG sources exist. One obvious data source are the websites of TV stations where typically program schedules are provided. Note, that automatic processing of this ”online EPG data” such as crawling is often forbidden, leading to rather poor availability and accessibility. Again, up-to-dateness and channel coverage can be mentioned as positive aspects, whereas data consistency, especially between different TV stations, is not present and therefore quite a negative aspect. Unfortunately, most of these schedules offer only very limited program descriptions. Thus, element coverage must be judged as rather poor.

External suppliers, also called internet EPG providers, are another important data source. Although, many open metadata standards exist, most of these suppliers provide data in a proprietary format. In most cases, the data consistency strongly depends on the specific external supplier and varies between medium and high. One of these suppliers is the Axel Springer AG with its service, epgData.com. It provides professionally prepared program information for more than 200 European channels with detailed program descriptions and a very extensive element coverage. Aside from standard elements like in DVB EIT, involved persons (actors, directors, studio guests, moderators) are listed as well as information about aspect ratio, audio, parental guidance, year of creation, country of creation, theme, detailed categorization information and preview images for the program. Data for upcoming programs is provided for up to 14 days and accessible via the internet. A drawback of this service is that the data is not freely available and that the up-to-dateness suffers due to the update time needed to get new data.

Table 2.2 summarizes the main pros and cons of each TV metadata source. Because of its extensive amount of data and its good channel coverage, we have selected epgData.com as the main data source used in this work.

2.2.8 Metadata Standard Selection

The selection of a proper metadata format for use in a specific application is a challenging task. Focusing on the TV domain, only a small subset of the available metadata standards must be examined. In the following section we will focus on a brief comparison of DVB-SI, MPEG-7, TV-Anytime, XMLTV and the BBC Programmes Ontology. For a structured comparison, we measure these standards based on the following criteria:

- Syntactical foundation: The syntactical foundation of most metadata standards is XML but plain text, binary formats, and semantic web technologies (RDF, OWL) are also present. It is important to examine this foundation because its complexities make necessary different levels of content processing.

	TV station (DVB)	TV station (internet)	epgData.com
Channel coverage	good (most TV stations provides EPG data)	good (most TV stations provide schedules online)	very good (all available channels)
Element coverage	medium	bad	very good
Availability and Accessibility	very good	medium	good
Consistency	bad - medium	bad - medium	medium - good
Up-to-dateness	very good	very good	medium - good

Table 2.2: Comparison of different TV metadata sources.

- Expressiveness: It describes how well a specific standard is able to express the required metadata for a piece of content and how well it allows for complex metadata description. Generally, a high level of expressiveness is desirable.

- Complexity: The complexity of a description and a metadata standard is often strongly related to its expressiveness. Because high levels of complexity usually result in the need for considerable amounts of processing, a trade-off between complexity and expressiveness must be found. Especially in the domain of TV we are confronted with very resource-constrained environments such as CE set-top boxes, and as a result, a well balanced trade-off is of high importance. Nevertheless, a complex but well aligned structure also helps to improve parsing and processing, compared to poorly structured or even unstructured descriptions.

- Coverage: Metadata standards commonly have a specific area of application and therefore a specific focus. For a standard in the realm of TV, its coverage should at least extend to the more general category of ”multimedia.”

- Development Status: An important characteristic of standards is their development status. Frequent changes of a standard due to a premature status lead to the demand for extensive changes in applications using this standard, and as a result, hurt interoperability. As TV metadata is at the foundation of iTV platforms and TV recommender systems, changes in metadata standards have to be considered as critical.

- Tools and Support: The application of metadata standards is often, depending on the standard’s complexity and expressiveness, a very challenging task. Thus, support and available tools like processing libraries, converters, database bindings, a detailed documentation and available specifications are important arguments for the application of a specific standard.

	MPEG-7	**TV-Anytime**	**XMLTV**	**DVB-SI**	**BBC Programmes Ontology**
Syntactical foundation	XML-based	XML-based (parts adopted from MPEG-7)	XML-based	plain text / table-based	RDF/OWL-based
Expressiveness	very high	high	low	low	medium
Complexity	very high	high	low	low	medium
Coverage	Multimedia in general	TV	TV	TV	TV
Development Status	mature	mature	well established	mature	experimental
Tools and Support	well supported	well supported	well supported	well supported	rudimentary support

Table 2.3: Comparison of different TV metadata standards.

Table 2.3 shows a comparison of the different metadata standards based on the criteria introduced above. As XMLTV and DVB-SI are, due to their low expressiveness (e.g. poor categorization), not applicable for sophisticated and complex program descriptions, they can not be considered as metadata standard in this work. The BBC Programmes Ontology offers a very interesting and innovative means for describing all parts of TV content and their relationships to one another. Nevertheless, this specification offers only a limited level of expressiveness and its current development status does not allow for its application. In this work we focus on the use of TV-Anytime as our basic metadata standard mainly because of its high expressiveness, its mature status and available tools. Featuring nearly the same expressiveness for program descriptions as MPEG-7, TVA is far less complex and more well established in the TV domain. Furthermore, it is strongly supported by the BBC which offers program descriptions for their schedule in the TVA format. Moreover, BBC Backstage[1], a open developer network for innovative TV related projects, provides an open source Java API[2] for representing, converting and processing of TV-Anytime documents.

[1] BBC Backstage – http://backstage.bbc.co.uk/

[2] BBC-TV-Anytime API – http://www.bbc.co.uk/opensource/projects/tv_anytime_api/

CHAPTER 3

Natural Language Processing

Natural Language Processing (NLP) is an umbrella term for all approaches to using computers to achieve human-like processing of natural languages. Generally, a ”natural language” is defined as a language that has developed organically over time in one or multiple societies. This definition excludes man-made languages, such as programming languages. NLP is a very active and interdisciplinary field of research with roots going back to the first attempts to use machines for breaking codes during the Second World War and for translation in the late 1940s. It is closely related to computer linguistics which can be understood as a connection between computer science and linguistics. Among others, NLP includes the following areas of application and research:

- Machine translation: Machine translation is primarily concerned with the process of automatically translating text or speech from one natural language into another. Approaches in this area range from elementary systems simply performing word substitution, to very complex, corpus based approaches, employing statistical, machine learning based or hybrid techniques.

- Speech recognition and synthesis: Processing of spoken language is accomplished using speech recognition and speech synthesis. Speech recognition is used to process spoken text and convert it into a computer processable form (e.g. written text). Text comprehension is a very active subfield within this research discipline. Typical applications of such mechanisms are the automatic conversion of spoken words to text (speech-to-text systems), speaker recognition, voice user interface and dialog systems. In contrast to speech recognition, speech synthesis aims to artificially produce human speech. It is often used for converting written text into speech (text-to-speech systems), dialog systems or to aid people with speech impairments.

- Information retrieval: In the field of information retrieval, NLP is used in almost all cases in which natural text is processed. NLP plays a significant role in the construction of concept- and meaning-based indices.

- Information extraction: Processes which attempt to selectively structure and combine recognizable information from different unstructured natural language documents are commonly grouped together as information extraction tasks. Their main aim is often to extract information about objects and their relationships in a specific domain.

> Thus information extraction is used for, among other things, text summarization, named-entity recognition, text categorization and classification. It is also used in other areas such as information retrieval.

For a detailed discussion of NLP and its application to Text mining, the interested reader is referred to [Jur00, Wit06].

In the following sections, we will concentrate on NLP techniques that can be used to split natural text into its parts (tokens) and on the extraction of additional information about the text and the words that compose it. Mechanisms and approaches discussed in this chapter are used in our enhanced tokenization component (cf. section 6.1.1). Thus, the level of detail is aligned with its application in our tokenizer, and is not intended to be a complete study of linguistics or computer linguistics. This chapter is structured as follows: First, the basic steps of text tokenization and high-level concepts concerning the structure of natural texts will be introduced in section 3.1. In section 3.2, several text processing methods from the morphological level of text, such as stemming and lemmatization, will be discussed. Section 3.3 considers the syntax of sentences, more precisely the categorical grammar used in Part-of-Speech (POS) tagging. Finally, section 3.4 covers semantics where several methods for extracting the meanings of words using lexical and semantic network languages are introduced.

3.1 Principal Elements of Natural Text and Tokenization

Tokenization is a term for the task of splitting text input into its constituent components. Thus, input text is provided to the tokenizer in its domain specific form, such as source code or natural text, to perform the task of splitting. It is used in many areas of application such as programming languages, security, speech recognition, information retrieval and natural language processing. A well known component used in compilers, the scanner, completes the tokenization step for programming languages. This step is also called lexical analysis. During lexical analysis the input text is read and split into a sequence of symbols. In the world of well defined languages, such as programming languages tokenization is often an easy task carried out by finite-state automatons.

The realm of natural languages, however, is often more complicated and usually requires a more inventive approach. Especially for natural language processing, proper tokenization is at the foundation of the whole process. The way textual input is divided in its components heavily influences the overall accuracy of NLP systems. Most problems that arise during this task have their origin in the basic character of natural languages. Although the definition of a token - a single word - seems to be clear, a common definition is missing. In the area of NLP there are many different notions of what counts as a token. Depending on the usage, tokens are commonly defined in many different ways and can refer to something as small as a byte or as large as a conjoined sentence. Figure 3.1 illustrates a simplification of such a hierarchy. On the lowest level, tokens can be interpreted as single bytes. One level above above this, bytes are combined and form a single character. On the next level are words, which are one of the most widely used definitions for tokens. In NLP punctuation marks, special characters such as ”@” or ”$” and numbers are often also interpreted as tokens at the word level. On a semantic level, words are combined to compounds in various forms such as ”Stacheldraht” (engl. barbed wire). Finally, whole sentences and conjoined sentences can also be defined as tokens.

Please note that a compromise between the granularity of tokens and the present semantics has to be found. Very fine grained tokens often lead to the loss of semantics (e.g. a date ”22.10.10” is considered on the character level) whereas high-level tokens such as sentences may negatively impact their processing steps and performance. In the following, we will concentrate on the token, as defined on the word level. The tokenization process can be divided into two major steps - the sentence segmentation and the word segmentation.

Sentence Segmentation

In this step, the tokenizer determines how the text can be divided into sentences. Most people can easily conduct this task based on their experience with a language and the sentence's semantic structure. In contrast to the relative ease with which a human can complete such a task, it is much more difficult to achieve the same level of accuracy with an algorithm. Punctuation marks such as periods, question marks and exclamation marks are commonly used to end a sentence. However, the reverse does not always hold. Punctuation marks are also used in abbreviations, to mark ellipsis or in URLs. In [Sta99] Stamatatos et al., the accuracy of sentence segmentation based on punctuation marks in different corpora has been evaluated. If a period is understood solely as an indicator for ”end of sentence,” about 53 % of the sentences in the Wall Street Journal corpus and up to 90 % in the Brown corpus would be identified correct. Aside from these basic approaches, other, more complicated methods have been proposed. These do not all treat periods simply as sentence boundaries, even when they are not preceded by abbreviations or when the next token is not capitalized. Many approaches are based on supervised machine learning techniques (cf. [Pal94, Rey97, Sta99]) or unsupervised machine learning techniques (cf.

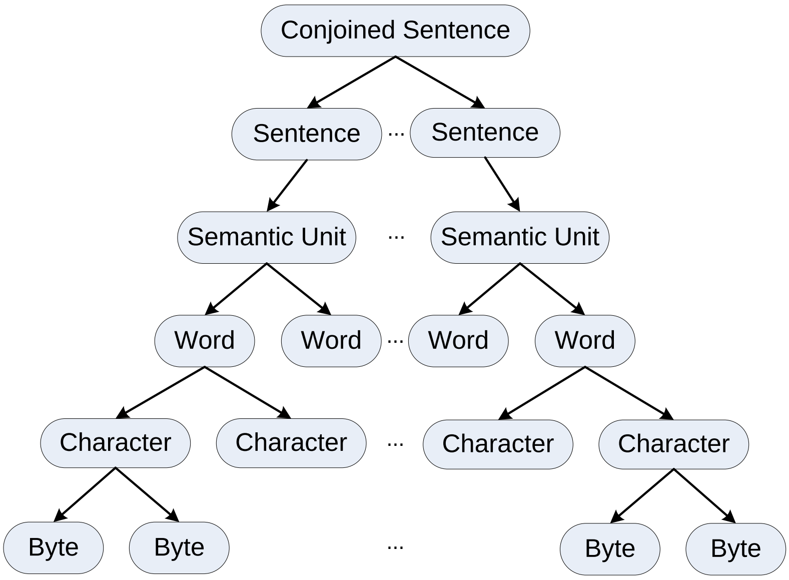


Figure 3.1: Simplification of an exemplary token hierarchy.

[Mik02, Kis06]). Current systems feature a precision of up to 99,5 %.

Word Segmentation

In this step, the tokenizer has to determine the word's boundaries. In many languages, such as English, most words are easy to separate by assuming that spaces indicate the end of one word and the beginning of another. Nevertheless, even in English contraction such as "I'll" or "don't," possessives ('player's'), foreign phrases such as "et cetera," numbers and URLs have to be handled differently. Contractions and possessives are commonly split into two tokens. In contrast to English or German, an accurate word segmentation of other languages like Chinese or Japanese poses a sizable research challenge, since a separator, like a space character between words, is missing. Interested readers are referred to the "Chinese Language Processing Bakeoff," in which Chinese word segmentation is one of the main topics.

Different tokenization methods have shown adequate precision in most languages, including German and English. With the exception of special areas of application such as biomedical literature or other heavily genre-specific text categories, the tokenization of most languages containing proper space characters or separators, can be viewed as solved.

3.2 Morphology

Morphology as a part of NLP covers the identification, analysis and description of word formation. Word forms typically change depending on the grammatical context. This is known as inflection. Inflections typically reflect different grammatical nuances, including case, tense, mode or gender. In general, the morphological analysis is based upon morphemes. Morphemes build the basic units of languages, and are often referred to as the smallest units providing semantic meaning. A morpheme can be a word, a word stem, an affix, a suffix, etc. Morphemes are divided into stand-alone and bound morphemes. As the name suggests, bound morphemes, in contrast to stand-alone ones, can not stand alone as a word and only make sense when combined with other morphemes. For instance, the word "unbreakable" is composed of three morphemes: The bound morphemes "un-" and "-able" and the unbound morpheme "break." In linguistics, words are often also referred to as lexemes. A lexeme can be understood as a combination of word form and meaning on a word level, a kind of "lexical morpheme." Different inflected forms of a specific word refer to the same morphological unit, the lexeme. Nevertheless, lexemes are abstract concepts with no intrinsic textual representation. In dictionaries it is common for lexemes of words - represented in a canonical form called lemma - to be listed without all of their derivational forms. For further processing of textual inputs, processing on the lexeme level allows for an easier treatment of words. Thus, in the following we will discuss different mechanisms used to treat words on a conceptual level, or to at least derive a common representation for different inflected forms of a word.

3.2.1 Stemming

Stemming generally describes a process that reduces inflected and sometimes derivational related forms of a word to its stem. The stem of a word should not be confused with its proper morphological root (e.g. the root of ”replacing” is ”replace”). In most cases a simple algorithm is applied that chops of the end of words, such as converting the word ”replacing” to ”replac.” This process is often governed by language dependent sets of rules. For special areas of application, domain-specific rules are also introduced. Due to the fact that natural languages do not possess completely regular structure, most stemmers inevitably make mistakes. Common errors are under-stemming and over-stemming. Under-stemming describes what happens when two words belonging to the same conceptual groups are mapped to different stems (e.g. ”create” to ”creat” and ”creation” to ”creation”). By contrast, over-stemming occurs when words belonging to different conceptual group are attributed to the same stem (e.g. ”generate” and ”generic” to ”gener”). Stemming algorithms can be grouped into the following categories:

- Brute Force: Employing a static look-up table with the inflected word forms and their related stems, the brute force approaches are able to determine the correct stem for each known word. One major concern facing this method is the the sheer amount of data needed in the look-up table in order to provide viable stemming results. Moreover, in such a system, continuous updating is needed.

- Affix Removal: This approach reduces a word to its stem by removing common pre- and suffixes (affix refers to both pre- and suffix). For this removal, a list of simple rules is used. Affix removal is a rather simple approach, because the list of rules is easy to maintain (for a linguist). Nevertheless, the quality of stemming is often poor. This is particularly the case with words that have irregular inflections (e.g. ”go,” ”went,” ”gone”).

- Probabilistic: Trained on a set of inflected word forms and their related stems, probabilistic stemmers are able to choose the most probable stem for a given word. With a proper training set, this approach is able to perform very well and can even do multilingual stemming. In most cases, getting a proper trainings corpus is the biggest challenge.

- Hybrid: In this approach multiple approaches are combined, such as adding a look-up table step for irregular inflections to an affix removal approach. Thus, the advantages of different approaches can be used and the overall stemming performance is improved.

Languages with a simple structure, precise language rules and few verb inflections are commonly well served by stemmers. By contrast, the stemming of languages such as Russian and Hebrew is still a challenging, and due to their complex morphological structure, an error-prone task. The first stemming algorithm was proposed by Julie Beth Lovins in 1968 [Lov68]. Today many stemmers are available for a variety of languages. Among them, English is one of the best supported languages. One of the most widely used algorithms for stemming English is Porter’s affix-striping algorithm [Por80]. For a detailed discussion and a rough comparison of different stemming approaches, the interested reader is referred to [Ful98].

3.2.2 Lemmatizer

A Lemmatizer reduces the number of inflected and derivational related forms of a word to its lemma. The lemma is the linguistically correct base form of a word (also called the dictionary form). The key elements of lemmatization are a dictionary and morphological analysis. The dictionary is often build as a look-up table containing the lemma for each word form (full-form-lexicon) or just the lemma together with a set of rules to create its inflected forms (base-form lexicon) [Vol00]. For the morphological analysis of a word, contextual information is needed. Thus a PoS tagger, as described in section 3.3, is often used to provide this information. On this basis, the linguistic differentiation of homographs, words with distinct meanings and origins, but are spelled the same, such as ”bow” (”to bow” and ”the bow”) is possible, and the correct lemma can be derived. In contrast to the stem, the lemma of each word is again a valid word in the given language. Thus, compared to artificial stems, lemmas are easily intelligible to humans. Unfortunately, lemmatization is more complex and therefore slower than stemming. Most approaches to lemmatization are language specific, because of the use of specific dictionaries and the set of rules. To reach a viable performance, most approaches are tailored to one specific language. More recent approaches employ machine learning techniques such as Ripple Down Rule learning [Pli08] or are based on Hierarchy of Linguistic Identities (HOLI) in [Ing08].

3.2.3 Stop Words

Extremely common words are sometimes referred to as stop or noise words. It is assumed that they do not specify or possess a significant meaning. Thus, those words are usually ignored or skipped in further processing steps. Common examples for English stop words are ”the,” ”a,” ”an,” ”be” and ”it.” In general, articles, conjunctions, prepositions, and pronouns are almost always used as stop words. Nevertheless, the handling of stop words should be done with care. Consider an information retrieval system in which a user would like to search for the famous Beatles song ”Let It Be.” This would be rather problematic if ”it” and ”be” were to be removed from the search as stop words. Today, the use of manually filtered and rather small stop word lists is common practice. These lists are commonly generated in a field and corpus specific manner by selecting terms that appear frequently in a specific text corpus. For instance, in the medical field the word ”patient” is likely to occur very frequent and therefore can be considered as a possible candidate for the stop word list. For most other fields this would not be true.

3.2.4 Compound Splitter

Compounding is a basic method of word formation in natural languages. Compound words are commonly thought of as words which are composed of at least two words with different stems e.g. the German word ”Aktionsplan” (engl. ”action plan”) a compound of ”Aktion” and ”Plan.” In general, only one part of these words define the case, number and the grammatical gender. It is called compound head. The German word ”Büroklammer” (engl. paper clip) is feminine and singular due to its head ”Klammer.” In many languages such as German, Dutch, Swedish and Greek, compound words are frequently used, and can be formed in very long and complex manners. For instance in German, words may be freely combined to form new words. This poses challenges for many NLP tasks, because the number of words is constantly increasing. In order to cope with this issue, the splitting of compounds is often needed for NLP applications such as machine translation, text classification and information retrieval.

Compound words are categorized into four types, based on their semantics and the compound heads role:

- Endocentric (or ’tatpurusa’): A headed compound whose head determines the syntactical function and the meaning of the whole word. On a semantic level, such compounds are specialized forms or subclasses of the class denoted by the compound’s head. The remaining words simply act as modifiers. For instance, ”bedroom” denotes a special kind of room. In English, most compounds are endocentric. According to Leonard Bloomfield, a compound can be classified as endocentric if the word has the same grammatical function as its head.

- Exocentric (or ’bahuvrihi’): Exocentric compounds are headless compounds where no element functions as the semantic head. In such a case, the meaning of the word as a whole can not typically be derived from any of its constituent parts. An example for an exocentric compound is ”lazy-bones.” Here lazy does not describe a characteristic of a pile of bones, instead it is meant to describe a lazy person. These compounds often stand in a ”has property” or ”has a” relation to the meaning of the whole word (cf. ”redskined” or ”blue-blooded”).

- Copulative (also co-compound or ’dvandva’): These compounds are composed of words with equal semantical status. No part of the word can be regarded as the head that determines the meaning of the entire word. Each part characterizes an individual aspect of the whole word and behaves as an independent constituent such as ”Nordwest” (engl. ”north-west”) where both directions north and west are equally important. In German and English this compound type is rarely used.

- Appositional: These compounds are made of equipollent parts which often have a similar meaning to ”X as well as Y.” For instance ”player-coach” means that someone is a player and a coach at the same time. Many appositional compounds have a close semantic affinity with copulative compounds. Thus, these words are often categorized as subtypes of copulative compounds.

For a detailed description of compound words and their types see [Kat93]. Please note that these words have to be handled with care because in many cases it is unclear if a word is a compound or not. For instance the germanized word ”teenager” may be accidentally interpreted as the compound ”Tee” (engl. ”tea”) and ”Nager” (engl. ”rodent”). For a human, this split might not make sense but in a rule-based approach an appropriate rule for such words is necessary. Moreover, many words can be split in multiple ways. In such cases, it is often unclear which is the correct way to split the compound. Consider the German word ”Wachstube” (engl. ”guardhouse”) which can be decomposed into ”Wach” and ”Stube” or ”Wachs” (engl. ”wax”) and ”Tube” (engl. ”tube”). In such cases, applying compound splitting in a careless way may lead to significant semantic errors.

Nevertheless, as the semantics of a compound are typically related to its constituent elements (except for exocentric compounds) the appropriate decomposition of compounds significantly improve most NLP applications. Especially in the case of ad-hoc generated compounds, these mechanisms are needed for the proper processing of these newly formed words. This step is of high importance in the area of machine translation. The evaluation of Braschler et al. [Bra04] suggests that in the area of information retrieval, de-compounding efforts improved retrieval results considerably.

3.3 Syntax

Syntax, in a linguistic sense, can be defined as the analysis and definition of the fundamental rules and principles for the structure of sentences in natural languages. In general, language rules are summarized by the term "grammar." Thus, syntax is often understood as a subset of grammar. Grammatical rules cover things like how words and sentences are constructed, the current grammatical category of a word in its specific context, and how words differ in different contexts. Recent research in this area is concerned with the relationships and possible mappings between natural languages in general and formal languages. Formal languages are languages which are specified by a well defined set of rules with no ambiguity (e.g. programming languages). Such a mapping would allow the definition of an universal translation mechanism based on this formal language.

In the following, we will concentrate on the categorization of words into parts of speech, such as verbs, nouns, etc. This step is also called PoS-tagging.

3.3.1 PoS-Tagging

Part of Speech (PoS) taggers are essential parts of most NLP systems, especially in areas such as speech recognition, information retrieval, information extraction or machine translation [Jur00]. A PoS tagger's main aim is to assign unambiguous tags to each word of an input text. These tags are often very detailed representations of a lexical (grammatical) category, also called the part of speech, of a word. They are defined in various tagsets, assigning a concrete meaning to each tag, such as the Stuttgart-Tübingen-Tagset (STTS) [Sch99] for German or the Penn-Treebank tagset for English. The main aim of such tagsets are to provide a detailed categorization for each word. Compared to the current ten lexical categories used in German [Hau00] the STTS defines 54 tags. Larger tagsets define more than 200 tags (c.f. TOSCA-ICE 270). The size of the tagsets varies heavily depending on the language, objectives and purpose for which they are designed. Due to the characteristic ambiguity of natural languages, PoS tagging is a difficult task. Many different approaches have been developed. In the following we will outline the most common of them.

Rule-Based Tagger

This approach originally proposed in [Kle63] and [Gre71] makes use of a set of established grammatical rules. In current implementations, more than 1000 rules are used. The process of rule-based tagging is made up of two phases. In the first phase, each word of the input text is tagged with all valid tags. In the next phase, grammatical rules are applied in order to reduce the list of tags for each word to a single tag. Compared to other approaches, a text corpus does not need to be preemptively annotated. On the other hand, the rule formulation is very difficult and requires expert knowledge (e.g. from a linguist).

Stochastic / Statistic Tagger

Stochastic taggers are based on the probability of different tags occurring and the fact that certain combinations of tags are more likely than others. A Hidden Markov Model (HMM) is often used to represent this information. Theses taggers are trained on a corpus which is usually manually tagged. Test sets are tagged according to the most frequent tag for each specific word in the training corpus. Thus, these taggers perform well on text corpora which are similar to the training corpus. However when confronted with ”new words” which were not present in the training corpus, the performance suffers significantly. Nevertheless, stochastic taggers commonly outperform rule-based approaches.

Transformation-Based Learning Tagger

The TBL tagger is also called a Brill tagger after Eric Brill whose original idea it was to combine the rule-based with stochastic approach. It makes use of a pre-tagged corpus and a dictionary with tag frequencies, to be used in training during the machine learning step. During this phase it introduces new transformation rules and adapts existing ones. The training step is then repeated until a certain threshold, such as a certain level of tagging accuracy, is reached as compared to the training corpus. In the case of ”new words,” this approach is able to use its set of rules to tag the words accordingly. One big advantage to this approach is, that the rules generated by the tagger can be inspected and, in case of an error, corrected by the user. In addition to superior precision, the execution speed of this approach is also about 10 times faster than a solely stochastic tagger, as long as the set of rules is compiled to a finite state transducer [Roc95].

Decision Tree-Based Tagger

Decision Tree-Based Taggers are very similar to stochastic taggers. Stochastic taggers are not able to distinguish between PoS combinations, that are impossible and those that are simply very rare and therefore not included in the training corpus. In order to be able to cope with rare combinations a very small positive probability is given to such combinations. Although this approach allows for the assignment of rare tag combinations, it may lead to a higher number of possible tagging errors and a longer computing time. To cope with this issue, taggers based on binary decision trees were introduced in [Sch94]. Such taggers feature very quick processing.

In the following listing, the sample output of a PoS tagger is shown. The tagger uses the common representation ***Token / PoS*** and the Penn-Treebank tagset (cf. [San90]).

```
Larger/NNP tagsets/NNS define/VBP more/JJR than/IN 200/NN
tags/VBZ ./SENT
```

In this output, ***NNP*** is used for *proper noun singular*, ***NNS*** for *noun plural*, ***VBP*** for *verb singular present non–3d*, ***JJR*** for *adjective comparative*, ***IN*** for *preposition/subordinating conjunction*, ***NN*** for *noun singular or mass* and ***VBZ*** for *verb 3rd person singular present*. ***/Sent*** marks the end of the sentence.

Current PoS taggers report precision of up to 98 % e.g. [Sch95] cites an accuracy of 97,5 % for German. Depending on the test, training corpus and the language in use, this percentage can vary considerably. Thus, the performance in real-world applications is expected to be much lower.

3.4 Semantics

Semantics is generally defined as the study of meaning. On a basic level, it is the meaning of signs, however researches in computer linguistics concentrate mainly on the task of understanding text. The level on which the meanings of words are studied is often referred to as lexical semantics. Several theories state that the meaning of a certain word is made up of its context and its relations to this context. Morphology as well as syntax provide the foundation for analyzing semantics.

In this section we concentrate on the semantics of linguistic units and their relationships to one another, such as synonymy, homonymy and polysemy. Additionally, topics such as the extraction of named entities and coreference resolution will be briefly covered.

3.4.1 Word Sense Disambiguation

In many cases, words can have multiple different meanings. This lexical ambiguity is often defined as a fundamental characteristic of natural languages. Based on data gathered from the lexical database WordNet, Miller et al. [Mil90] discovered that the 121 most frequent English nouns have 7.8 different meanings on average. For humans, a text often contains rather little ”real” ambiguity due to broad background knowledge and contextual understanding. Nevertheless, even humans are sometimes unable to determine the intended meaning of a word. In computational linguistics this task – to determine the sense of a word in its particular context – is known as Word Sense Disambiguation (WSD). Words which are spelled the same, but unrelated in meaning and origin are called ”homographs.” For instance the word ”bank” can be understood to mean a financial institution or a riverside. On a finer level, ”bank” can be understood in several closely related senses, such as the company, the building itself or a reserve of money. The distinctions between this second set of meanings are not as drastic, or as easy to define as those in the first example. Figure 3.2 shows a visualization of the different meanings of the word ”bank,”as generated by Visuwords[1] and based on a dataset from WordNet.

WSD research began in the late 1940s. Although in many cases the word’s lexical class is sufficient for determining the correct meaning of words, many different comprehensive approaches for WSD have been proposed in recent years. These approaches are generally categorized according to the main source of knowledge used in the process of disambiguation:

- Dictionary or Knowledge-based: Approaches in this category primarily use dictionaries, thesauri and lexical knowledge bases, such as WordNet (cf. [Ban02]) for WSD. A trainings corpus is not used in these approaches. The most likely meaning for a word in a given context is identified based on a measure of contextual overlap between dictionary definitions (cf. [Les86]) , semantic similarity among concepts in semantic networks and by using heuristic methods such as assigning the most frequent sense.

[1] Visuwords – http://www.visuwords.com/

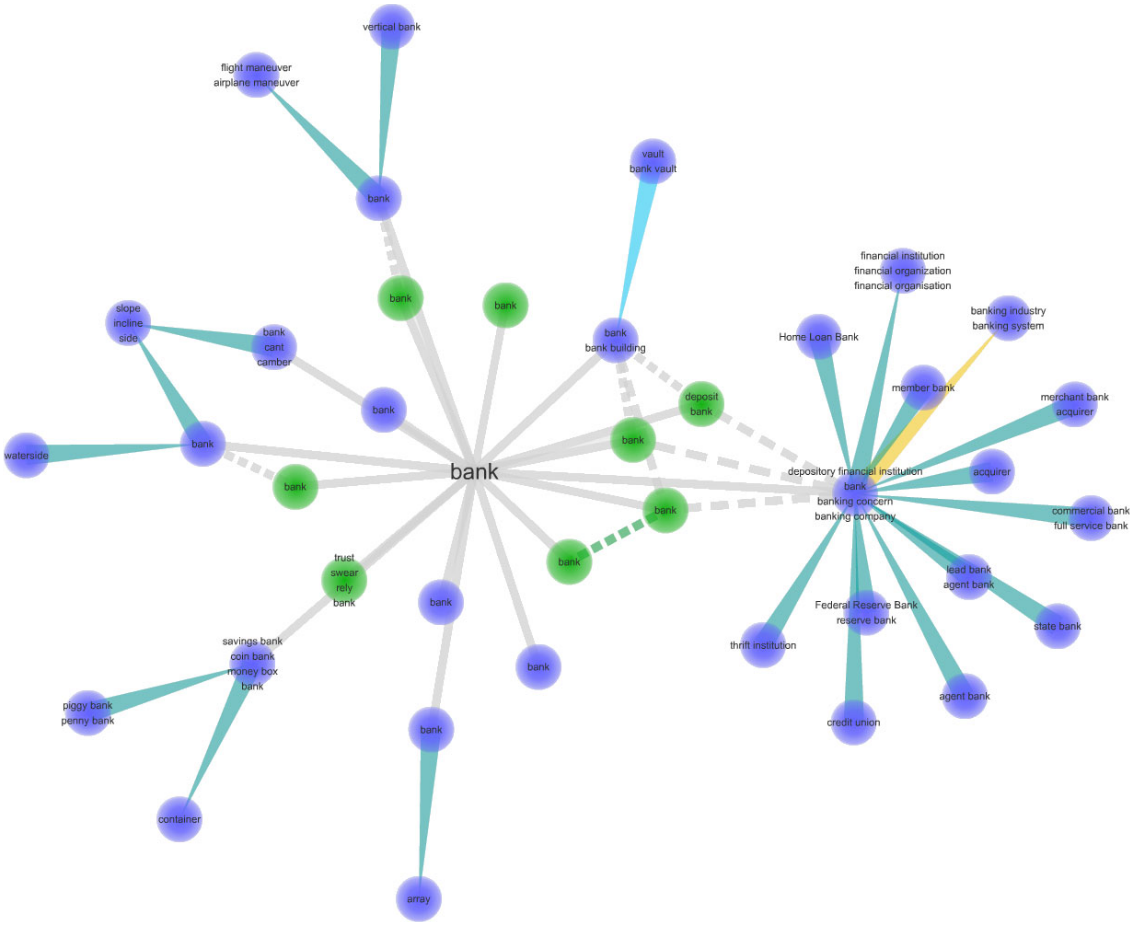


Figure 3.2: Visualization of the WordNet for ”bank” generated with Visuwords.

- Unsupervised corpus based: Here, unannotated corpora are used. Approaches in this category can be roughly divided into distributional and translational approaches. Distributional approaches try to find clusters or groups with similar contexts (or occurrences) where the given word is used in a particular meaning. Translational methods make use of the fact that ambiguous words in a source language are often translated as completely different words in a second language. Through the use of cross-language distinctions such as these, multiple meanings of a word may be uncovered.
- Supervised corpus based: These approaches are normally trained on a manually tagged corpora. Machine learning and statistical classification methods are then applied to determine the meaning of a word in a particular context.

Recent methods combine different existing techniques to overcome individual limitations. Evaluations of different state-of-the-art WSD systems have shown that supervised methods feature superior performance compared to unsupervised and knowledge-based approaches. Nevertheless, knowledge acquisition is still a bottleneck for supervised WSD systems. Current approaches such as [Mih07] try to overcome this problem by using encyclopedia such as Wikipedia as annotated copora. For a comprehensive discussion of WSD, the reader is referred to [Agi06].

3.4.2 Lexical and Semantic Network for Languages

Lexical and semantic networks for languages, commonly called word nets, are very similar to ontologies. Some sources even refer to word nets as language ontologies, although this label is not fully accurate. In general, an ontology is a formal conceptualization of a specific part of the world. Furthermore, it defines a set of explicit constraints describing the axioms of the world. In other words, it describes the existing concepts, their properties and their relations in a specific field. The complexity and size of an ontology depends primarily on the field covered and the chosen level of detail. Thus, most ontologies are build for very specific fields of knowledge. Usually they are used as a fundamental agreement between different parties in a communication process preserving consistency in usage of concepts. Elements common to most ontologies are: ”concept,” ”relation” and ”instance.” Today, ontologies are used for representing knowledge in many areas, such as software engineering, biomedical software, artificial intelligence and the Semantic Web. Prominent examples of domain specific ontologies can be found in the medical and biomedical field, such as the Gene Ontology (GO) and the Unified Medical Language System (UMLS). Other examples, however, are community driven, such as DBpedia[1] and Common Tag (ctag)[2]. In NLP, ontologies often occur in the form of taxonomies and thesauri. Ontologies consisting solely of subtype-supertype relations and instances are called taxonomies. A thesaurus, as a further specialized resource, only contains of synonymy relations.

In the following section, we focus on word nets as source for expressing lexical structures and semantics. Word nets are composed of synsets, and the various conceptual, semantic and lexical relationships between them. Words are grouped into semantically equivalent sets called synsets. They represent a word’s specific meaning, in the form of a group of synonyms. Most word nets are able to express the following lexical and semantical relationships, whereas the precise semantics of the relationships is not exactly defined or agreed upon:

- **Synonymy**: Synonyms are words which have very similar meanings and therefore can be substituted for one another in a sentence without altering the intended sense of the word within it’s given context. For instance, ”car,” ”auto” or ”automobile” can be used synonymously.

- **Antonymy**: It defines a lexical relationship between word forms where one word is the opposite of the other. For instance, ”rich” is the antonym of ”poor” and vice versa. Usually, an antonym for a specific word can be formed by adding a ”not” before the word. For some words such as ”rich,” where ”not rich” is not a valid antonym, this rule can not always be strictly followed.

- **Hyponymy / Hyperonymy**: It defines a ”is-a” or ”kind-of” relationship between synsets. Commonly, the concept of synset $\{x_1,x_2,\ldots\}$ is a hypernym of $\{y_1,y_2,\ldots\}$

[1] DBpedia – http://dbpedia.org/

[2] Common Tag – http://www.commontag.org/

when people agree that ”an x is a y, or a kind of y.” In this case, x and y are in a specialization / generalization relationship. For instance, a Koi carp is a hypernym of fish or animal. Hypernymy is a transitive and an asymmetrical relationship.

- **Meronymy / Holonymy**: These relationships are often called ”part-whole” or ”Has-A” relationships, and are both transitive and asymmetric. The concept of synset $\{x_1,x_2,\ldots\}$ is a meronym of $\{y_1,y_2,\ldots\}$ when people commonly agree that ”an x is a part of y.” This relationship is frequently used to construct a part-of hierarchy. For instance, a ”finger” is a meronym of ”hand.”

- **Troponomy**: A troponym is a verb expressing a specific manner of doing something. It encodes different elaborations of the base verb. For instance, troponyms of ”see” are ”gaze” or ”stare.”

- **Entailment:** This is the case when one verb either entails or necessarily implies another. For instance, ”snoring” (V_1) entails ”sleeping” (V_2). V_1 cannot be done unless V_2 has been done. Troponyms and Entailments are closely related as a troponym is a particular kind of entailment.

- **Causation**: In this relationship, one verb can always be seen as the cause and the other as the result. For instance ”see” is in a causal relation to ”show.”

A detailed discussion of these relations can be found in [Mil90].

Figure 3.3 shows a subset of a word net graph for the word ”Hummer” and its meanings as a car, auto brand or as a kind of singer. The node ”entity” is introduced as an artificial root of the hyponym tree. The most prominent word net resource for English is WordNet[1], developed by the University of Princeton. Other prominent resources are GermaNet[2] for German from the University of Tübingen, and for multiple European languages, EuroWordNet[3], which was developed within an European research project. According to Gomez-Perez & Benjamins [Per99], Wordnet can be categorized as top-level taxonomy, describing very general concepts without being limited to specific fields. WordNet is available in version 3.0 and offers approximately 146000 nouns organized in 118000 synsets, 25000 verbs in 12000 synsets, 30000 adjectives in 22000 synsets and 5600 adverbs in 4500 synsets. Additionally, for each grammatical category a separate taxonomy exists.

On a formal level, the structure of a word net can be described using a edge-labeled multi graph G. This graph is made up of a finite set of synsets (S) and, depending on the semantic relations defined, various interconnections between them. E is the union of all different edge types (R) representing relationships such as Antonymy, Hyponymy, Meronymy, Troponomy, Entailments or Causations. Compared to the Princeton WordNet, other more specialized language networks often include relationships, to model field-of-knowledge specific constraints and relations.

[1] Princeton WordNet – http://wordnet.princeton.edu/

[2] GermaNet – http://www.sfs.uni-tuebingen.de/GermaNet/

[3] EuroWordNet – http://www.illc.uva.nl/EuroWordNet/

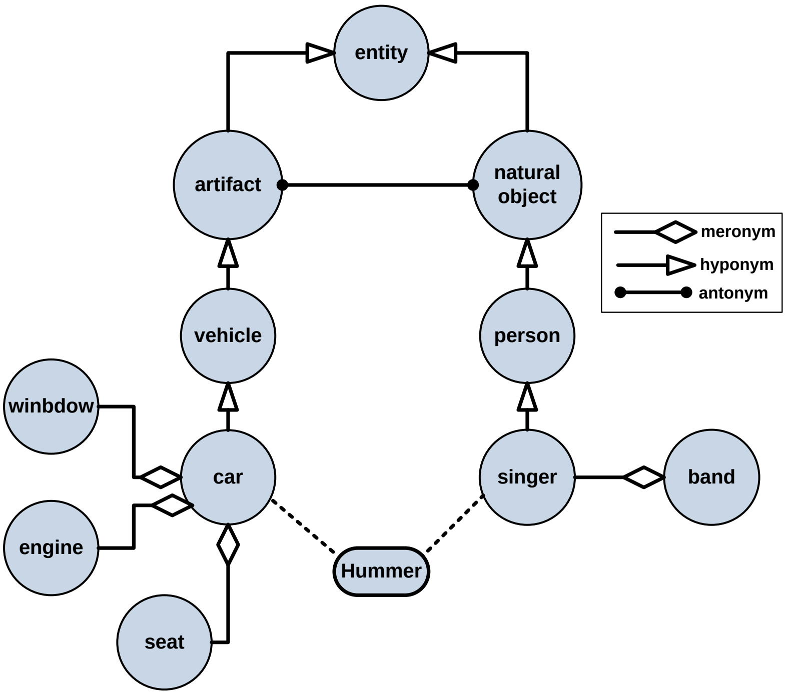


Figure 3.3: Subset of a word net graph.

$$\begin{gathered} G = (S, E) \text{ with } E := \bigcup_{R \in \mathcal{R}} R \\ \mathcal{R} = \{R_1, \ldots, R_N\} \text{ and } R_i = \{r_i | r_i = (u,v) \in S \times S\} \end{gathered} \tag{3.1}$$

Ontologies are often represented by directed graphs, whereas word nets are clearly undirected due to symmetric relationships such as Antonyms. G is generally said to be acyclic, as long as no synset is defined to be synonymous to itself. Word nets are often used to measure semantic relatedness. In the following, the most common approaches for measuring this relatedness will be described. A rough differentiation can be made between structure-based approaches such as the Leacock-Chodrow [Lea98], the Wu-Palmer [Wu94] measure and information-based measures such as the Resnik [Res95] and the Lin [Lin98] measure. The main difference between these variants is that structure-based measures rely solely on basic structural properties, such as depth and distance in the word net or parts of it, whereas information-based measures also include derived information, such as occurrence probabilities and frequencies. For a detailed discussion and evaluation of different semantic relatedness measures the interested reader is referred to [Bud06, Pat03]. In the following, we will first define some measures and mechanisms common to most semantic relatedness approaches. Afterwards, different relatedness measures will be briefly introduced.

Definition of Distance

The distance ($dist(c_1,c_2)$) between two concepts $c_1 \in S$ and $c_2 \in S$ is defined as the smallest number of steps (minimum number of intermediate concepts) on the shortest path between the two concepts and a common concept c. In the worst case the only common concept on the path is the artificial root T of the word net graph.

Definition of Depth

The depth of a concept $c \in S$ is commonly defined as the distance between concept c and the root node $T \in S$ of the word net graph. Because of this, it can be defined with the help of $dist$ as follows:

$$depth(c) := dist(c,T) \tag{3.2}$$

Definition of Information Content (IC)

Information Content (IC) is a measure of the specificity of a concept c. It is defined by the negative log of its occurrence probability $p(c)$ in a text. This probability is determined by the maximum likelihood estimation (MLE) of the concept's frequency in a large corpus. IC is defined as follows:

$$IC(c) = -\log p(c) \tag{3.3}$$

Definition of Lowest Super-Ordinate (LSO)

The Lowest Super-Ordinate (LSO) of two concepts $LSO(c_1, c_2)$ is defined by the lowest common subsumer of both concepts in a "is-a" hierarchy. It measures the amount of information the two concepts share. For instance, the LSO of the concepts "nurse" and "doctor" in the medical sense would be the concept "health profession" (depending on the is-a hierarchy in use).

Leacock-Chodrow Measure

The Leacock-Chodrow (LC) measure determines the similarity between two concepts based on the length of the path between the concepts in the is-a hierarchy. It assumes that similar concepts are close together in the hierarchy. The LC is defined as follows:

$$sim_{LC}(c_1, c_2) = -\log \frac{dist(c_1, c_2)}{2 \times \max\limits_{c \in S}(depth(c))} \tag{3.4}$$

Wu-Palmer Measure

The Wu-Palmer (WP) measure is based on the edge distance of two concepts. In this measure, the most specific subsumer of both concepts is also taken into account. Similar to the LC, it only considers hyponymy and hyperonymy.

$$sim_{WP}(c_1, c_2) = \frac{2 \times depth(LSO(c_1, c_2))}{dist(c_1, LSO(c_1, c_2)) + dist(c_2, LSO(c_1, c_2)) + 2 \times depth(LSO(c_1, c_2))} \tag{3.5}$$

Resnik Measure

Resnik defines his measure of similarity based upon the the amount of information that two concepts share with the IC of the LSO. In this measure he introduces the definition of IC as defined above.

$$sim_R(c_1, c_2) = -\log p(LSO(c_1, c_2)) \tag{3.6}$$

Lin Measure

Lin defines a semantic similarity theorem for his measure as follows: ”The similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are.”[Lin98]. Lin incorporates the IC of the LSO and the IC of both concepts in his measure as follows:

$$sim_L(c_1, c_2) = \frac{2 \times \log p(LSO(c_1, c_2))}{\log p(c_1) + \log p(c_2)} \tag{3.7}$$

3.4.3 Named Entity Recognition

Named Entity Recognition (NER), as a computer linguistics task, seeks to extract and accordingly classify tokens or token combinations into several predefined categories. Commonly used categories are persons, locations, organizations, expressions of time, percentages and monetary values. As a subtask of the information extraction domain, it is also often called Entity Extraction or Entity Identification. Most NE systems mark NEs by assigning tags to tokens and groups of them. The following listing shows an example where NE’s have been marked with SGML tags defined in the rather small ENAMEX tagset of the Message Understanding Conference - 7 (MUC-7).

```
<ENAMEX TYPE="PERSON">Peter Wolf</ENAMEX> vice-president of <ENAMEX
TYPE="ORGANIZATION">BMW</ENAMEX> announced the launch of the new
production line in <ENAMEX TYPE="LOCATION">Berlin</ENAMEX> <TIMEX
TYPE="DATE">next Monday</TIMEX>.
```

Other, more detailed tagsets such as the Sekine’s or BBN tagset specify up to 200 tags for this task. The progress of NER has been greatly advanced by the MUC-6 and MUC-7 organized by the Defense Advanced Research Projects Agency (DARPA) in 1995 and 1997. MUC-7 reported an f-measure of up to 97 % for NER evaluated on a given text corpus with a specific domain. Although 97 % seems to be an impressive value, the performance varies considerably depending on the given text corpus and its characteristics. Since MUC-7, many different approaches have been developed to further enhance the performance of NE systems. The NE systems can be roughly categorized as follows:

- Dictionary-based approaches: These approaches make use of comprehensive NE dictionaries and gazetteers (geographical dictionaries). Each NE is explicitly mentioned in connection with its related category. By comparing tokens and token groups with the dictionary entries, a NE Tag can be assigned. Due to the huge number of NEs and the limited size of dictionaries these approaches are constrained in their NE extraction capability. The occurrence of new NEs is another big problem for this approach. Solely based on dictionaries, ambiguous words cannot be identified and therefore not reliably recognized such as ”Paris” which can be a NE of type person or location.

- Rule-based approaches: Many NE systems make use of manually built rule sets, consisting of context-sensitive reduction rules (cf. [Kru98, Bla98]). For instance, the rule ”*Title Capitalized Word* $\Longrightarrow$ *Title Person name*” could be used to extract NEs of type ”PERSON.” Although these approaches are relatively simple they require expert knowledge for the rule creation. Typically these rule-sets are extended with many exceptions describing when a specific rule should not be applied. Moreover, these rule-based approaches are very error-prone in cases where new NE patterns occur.

- Supervised corpus based approaches: In this category, machine learning techniques such as decision trees, support vector machines ([Asa03]), hidden Markov models ([Bik97, Kle03] or approaches based on maximum entropy models ([Bor98, Chi03]) are applied on a pre-tagged training corpus. Although these approaches feature a very good NER performance, the data acquisition for the training corpus is expensive.

- Unsupervised corpus based approaches: Faced with the challenge of data acquisition, unsupervised systems try to extract NEs without the need for a prior labeled training corpus or manually constructed dictionaries. Most systems in this category start with a set of statistical or heuristic rules for extracting and classifying NEs of a large text corpus. For instance, in [Nad06] web search engines are used to build this unlabeled corpus. Based on the context of the NEs, new rules are introduced. In this manner, a dictionary for NER can be generated and used.

State-of-the-art NER Systems combine different approaches into semi-supervised systems (cf. [Nad07a]). In most systems, resources such as gazetteer or simple rule-sets and preprocessing steps (e.g. PoS tagging) are also introduced. For a comprehensive overview of different NER approaches, see [Nad07b]. However, most of these approaches concentrate on a certain domains and specific textual genres, such as news articles or web pages.

3.4.4 Coreference Resolution

Coreference resolution (CR) is the process of identifying and grouping expressions in a natural language text that refer to the same logical entity in the world. The entity referred to is often called referent or antecedent. More formally expression 1 ($exp1$) and expression 2 ($exp2$) corefer if and only if $referent(exp1) = referent(exp2)$. References are often divided into two main types:

- Exophoric references: Exophoric means that an entity, which is not directly mentioned in the text, is referenced. It also means that for human readers, it is not possible to identify the referent without further contextual knowledge. Something in the extralinguistic environment is referenced.

- Endophoric references: These references point to a referent that is mentioned in the text and can be easily identified based solely on the context. Depending on the referent's position in relation to the reference, this type is further split into anaphoric and cataphoric. Anaphoric means that the referent has been previously mentioned and cataphoric means the referent has not yet, but will later appear in the text.

For a better understanding of coreferences consider the following example.

```
Schwarzenegger arrived at the airport.  He was welcomed by an official
delegation.  Afterward the governor went to his hotel.
```

With adequate background knowledge it is easy to see that ”Schwarzenegger”, ”He” and ”Governor” refer the same entity. ”He” in the second sentence is a typical example for an endophoric reference. Whereas ”that” in the statement ”look at that,” as said by someone pointing at an image, is an exophoric reference. In many cases, even in simple sentences, the referent can not be determined with 100% certainty. Consider the following examples:

```
The child grappled with the dog.  It barked.
The child grappled with the dog.  It laughed.
The child grappled with the dog.  It was happy.
```

In the first two sentences it is easy for a human to determine the correct referent based on proper background knowledge. By contrast, in the last sentence it is unclear if ”It” refers to the child or to the dog.

Especially in dialogs, references are often repeated and form so called ”coreference chains.” In order to draw proper conclusions about the referent the subsequent referential forms of the chain must be traced back to the original reference.

All of these elements contribute to the complexity of the CR task. Much research has been conducted in this area. Statistical and machine learning approaches yield outstanding results [Ge98]. Similar contributions came from McCarthy and Lehnert [McC95], Soon et al. [Soo01] and Ng and Cardie [Ng02]. Common features of semantical, syntactical and lexical levels are taken into account in the previously mentioned approaches. Among many PoS tags, gender, animacity, minimum-edit-distance (MED) [Wag74] and the grammatical function is used. In [Str02] different MED based features have been evaluated and compared. In [Yan03, Luo04, Dau05] several complex approaches for modeling coreferences in a global discourse have been proposed. Recent research attempts to include semantic information from different sources into the coreference models. For instance, in [Har01], WordNet and in [Pon06] Wikipedia is used as a source of knowledge. Additionally, coreference pattern harvesting is often applied (cf. [Yan07, Mar05]).

Although several approaches and implementations for CR exist, their application in real world situations has yet to be evaluated in detail. CR approaches are often either language dependent or focused on a specific application domain. The assumption of readily available features for CR is also often a problem in current applications, as the extraction of such features can be very complex and error-prone.

CHAPTER 4

Recommendation Systems

In every days life we frequently have to reach decisions and make choices. We have to narrow down choices on everything from cars, jobs and products, to movies and so many other things. Often we do not have a rich knowledge base to aid in making the ”right” choice. A natural and social way to cope with this situation is to ask other people for help making a selection and for their recommendations. With important decisions, we often consult professionals such as a real estate broker, a placement officer or a car salesman. Generally, recommendations can be divided into personalized and neutral recommendations. Personalized ones are directly targeted at and adapted to an individual user, similar to the recommendations of a friend. Reviews and rating in magazines, on the other hand, can be seen as neutral recommendations. Another way to improve this decision making process has emerged with the world wide web, which provides both personalized and neutral recommendations. People can search for reviews, tests, discussion forums, further information and ratings, or simply discuss their choices within their social network.

Recommender systems are designed to assist the user in his or her social decision making process. Going back to the roots of these suggestions, recommender systems were defined as approaches where ”people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients” [Res97]. Over time the connotation of the term ”recommendation system” has changed. Today, a widely accepted description of the same term is as follows: ”Recommender systems form a specific type of information filtering (IF) technique that attempts to present information items (movies, music, books, news, images, web pages, etc.) that are likely of interest to the user.” [Zho09]. The main difference between the former and the latter definition is that in the latter one, personalized recommendations, are primarily seen as an output product of the recommendation system. One of the main aims of these systems is the recommendation of specific items or services a user is most likely interested in, or to predict a rating for these items. Thus, the main recommendation function is often divided into the following two categories:

- Recommendation of items: In recommendation-based approaches, a list of items, considered to be useful for the user by the recommendation system, is presented to him or her. The items are ranked based on a value measuring their usefulness. Often this value is measured by taking scores, such as the average preliminary ratings of the item, into account, or by predicting a score value that should reflect the current user’s rating for the item. In many applications, the item count of the recommendation list is limited to the N highest ranked items. Thus, this approach is commonly called Top-N recommendation.

- Prediction of ratings: Approaches that try to estimate how a user would rate a specific item are often referred to as rating-based systems. Typically, the rating is predicted based on the ratings of other (similar) users.

Even though recommendation- and rating-based systems are very similar, they differ in levels of complexity and effort. The latter approach must be able to predict ratings for all available items, whereas the first only needs to offer a limited number of recommendations (typically a list of N). Particularly in cases of items that are rarely connected, or that share only a few similar users, prediction can be a very challenging task.

Although recommendation precision and quality are important factors for recommender systems, their success is mainly determined by user acceptance. Today, recommender systems are used in various fields, such as online shops, intelligent software agents, news portals, and music and video platforms. One of the most prominent examples is the online marketplace, Amazon, with its recommendations such as ”People who bought this item also bought ...” or ”People who where interested in this item were also interested in ... .” It is commonly believed that up to 20 % additional revenue is generated solely by Amazon’s recommendations system, although official numbers are part of Amazon’s business secrets.

In general, recommendation systems are categorized into content-based, collaborative and hybrid systems based on the type of mechanism used for generating these suggestions. In the following, we will introduce a detailed categorization of recommendation systems (cf. section 4.1), allowing for a better differentiation between approaches. Then, we analyze the main categories of recommender systems – the content-based systems in section 4.2 and in section 4.3 the collaborative systems. Both recommendation approaches have been used in our TV recommendation system. Finally, section 4.4 will discuss different ways to combine the aforementioned methods in the form of hybrid approaches.

4.1 Categorization

Although, most sources categorize recommender systems solely based on their recommendation generating mechanisms, several other dimensions exist that can be taken into consideration when differentiating between systems. In the following section, the dimensions of data collection, the level of identification and the degree of personalization will be introduced. This categorization has been, to some extent, adapted from Andreas Neuman [Neu09] and Resnick et al. [Res97].

Data Collection

One of the most obvious distinctions can be made based on the way data is entered into and collected by the system. Commonly, this data is used as the foundation of user profile formation. The following methods of collecting data are frequently used:

- Implicit: Implicit data is typically collected by monitoring the user and his or her behavior. Depending on the area of application, either the viewing times and/or the interactions of the user with the system are analyzed. In this step, items either purchased or consumed are taken into account. Based on the user's actions, similarities to those of other users can be measured and used in later recommendation steps. Note that this data can be easily collected and used without any extra effort on the user's part. Unfortunately, implicit data is often hard to interpret. Questions, such as "What conclusion may be drawn from a specific viewing time in relation to the user interests?" or "What is the decisive attribute of a movie (e.g. the genre or the director) that motivates a user to watch it?" are hard or even impossible to answer.

- Explicit: Explicit data is collected using direct feedback from the user. This is often done by asking the user to rate, rank or annotate items. Typically requests such as "Please give a rating from one star to five stars for the movie" are used to collect this data. It provides very good, high quality feedback that can be used to build user profiles. For the most part, it is easier to interpret than implicit data. Nevertheless, by simply rating an item such as a movie, questions regarding the decisive attribute for the user's opinion of the movie remain unanswered. Another drawback is that the user has to be bothered with entering the data.
- Hybrid: One way to combine advantages of both aforementioned methods for collecting data is the hybrid approach. In general, implicit and explicit methods are used together to varying extents. Some systems use explicit methods to gather a basic profile (e.g. demographic information of the user) and use, for advanced profiling, the implicit method. Other systems use implicit data simply as a kind of add-on to an explicit profile. The split between the two methods is often done in a domain and application specific way.

Degree of Personalization

Another important element is the recipient of the recommendations and their degree of personalization. We distinguish the following categories:

- Individual: Recommendations are generated for and addressed to individual users. The generation of recommendations is is based solely upon the user's likes and dislikes.
- Group-centered: With this method, users are grouped based on similar characteristics and attributes. For instance, such a group might be using demographic attributes, such as considering a group of people from the city of Passau that are between the ages of 20 and 30 years old. Recommendations are then generated for the whole user group.
- Mixed mode: In mixed approaches, the two modes are combined. Group recommendations are supplemented with individual ones in these approaches.

Identification Level

In order to gather information about people's preferences, some kind of identification mechanism is needed. Identification is often done on different levels. Additionally, the location of this identification is also of great importance. For example, is the identification done on an external server or just locally in the user's set-top box? In the following section, we will discuss these important levels:

- Anonymous: On the anonymous level, recommendations are generated without identifying the current user. It is often used on a session or transaction basis, which means that only a small amount of information from the user's interaction with the system can be used to generate recommendations.

- Pseudonym: A user (or sometimes a group) is tagged or associated with a source of identification, the pseudonym. The real identity of the user is hidden by his or her pseudonym. Often, also a kind of avatar is used as a representation of an individual user, or for the user as a part of a group, such as a family. On this level, detailed analysis and recommendation generation is possible based on a well-rounded view of the interaction of the user with the systems.

- Full identification: The user is identified with his or her real identity. This level offers comprehensive ways of incorporating personal information about the user into the recommendation generation process. Nevertheless, this level often raises privacy issues and is refused by many users. In this category, the location of the identification is of particular importance.

Today most recommendation systems make use of the pseudonym level.

4.2 Content-Based Approaches

Recommendation systems and information retrieval and filtering systems are very similar in terms of the way they function. Both systems attempt to find the most relevant items for a user or group of users when faced with a large collection of available items. Because of this, content-based (CB) recommendation approaches are often treated as a particular type of information filtering system, and referred to as content-based filtering approaches. The roots of content-based filtering approaches go back to the field of information retrieval and information filtering in the early 1990s [Ado05]. Based on this background, most content-based recommendation approaches still focus on textual information. One example of the application of these mechanisms is the personalized web radio system Pandora[1]. The combination of content-based and collaborative approaches is particularly common in Web shops (cf. section 4.4).

In CB systems, content is not clearly defined and may refer to different things. For instance, in video recommender systems, it may refer to the video itself, to descriptive metadata (title, actors, etc.) or to both content types. Web recommender often use the content of webpages, the URLs, the anchor texts to them and other metadata about the website. Free of a concrete definition, content-based approaches discover items of interest solely by analyzing their content. In most cases, the user's history serves as the foundation for acquiring knowledge about his or her interests and preferences. By analyzing 'the items' content, the system tries to identify commonalities between items in the user history and

[1] Pandora – http://www.pandora.com/

other items in the collection. Accordingly, a score is calculated by the recommendation engine and used as an indicator for measuring the similarity of a current item to items the user liked in the past. In these systems, no information about other users, their relation to the current user or their interaction with the system is employed. [Paz07]

According to the work of Adomavicius et al. [Ado05], content-based approaches are grouped based on which fundamental mechanisms they use. On one side, heuristic approaches make use of basic mechanisms from information retrieval and filtering. One of the most prominent examples is the use of the vector space model [Sal86] to represent documents (items) and the user profiles as term vectors, and to measure the similarity between them using a function such as the Cosine similarity (cf. equation (4.1)).

On the other side, model-based approaches frequently use methods of machine learning and artificial intelligence, such as artificial neural networks, support vector machines and Bayesian classifiers. These approaches employ a user model to measure the relevance of an item for a specific user, and the likelihood that the user is interested in that item. This process is often referred to as classification. In the most basic variant, we simply distinguish between ”interesting” and ”not interesting” (a binary classification task) for a specific user. Based on the user history, a user model is derived in a learning step. Most classification approaches allow the use of explicit and implicit data for building this user model. In the following work, we focus on model-based CB approaches. All approaches used are discussed in detail in chapter 6 section 6.1.

Challenges

Although CB filtering mechanisms are able to provide good recommendations, in most areas of application, several limitations should be mentioned:

- Content Analysis: One of the most obvious problems is caused by the content itself. If items of interest and other items can not be distinguished, due to missing information or bad content quality, no adequate recommendations can be generated. Moreover, the representation of items in the recommendation process must also be done in a way, that ensures that different items are distinguishable. Thus, content quality, quantity and the representation of items are key features to success.

- New User Problem: Upon the introduction of a new user into the system, no user history and no information about his or her interests and preferences are available. Because a proper description of the user with at least some indication of the user’s interest is needed to generate recommendations, no items of interest can be immediately identified. A common way to cope with this problem is to ask new users directly in the registration step of the system for their preferences. As a result, a basic initial profile of the user is available.

- Overspecialization Problem: ”Overspecialization” is a typical problem CB filtering systems face. It is a way to describe the problem that occurs when only items with a high similarity to the user’s profile are considered for recommendation. Although this may sound quite natural for a recommendation system, it leads to a very narrow recommendations and hardly any diversity in items selected as recommendations. Items that are not similar to anything the user liked in the past would not get recommended. Nevertheless, a recommendation system should provide a wider variety of diverse items, including some ”unexpected” items the user may be interested in, but would not choose by himself without the help of the system. A basic way to guarantee diversity is to introduce some randomness into the recommendation process. On a more systematical level, Abbassi et al. [Abb09] introduced a trade-off between relevance and risk of recommendations, based on the identification of item groups that are underexposed in the user's profile.

4.3 Collaborative Approaches

Collaborative approaches are commonly referred to as ”collaborative filtering” systems. This term was coined by Golberg et al. who introduced Tapestry, a collaborative email filtering system, in [Gol92]. Today, a huge variety of systems in different domains make use of collaborative filtering (CF) approaches. Well known commercial examples for the application of these mechanisms are Amazon[1], Ebay[2], Netflix[3] and TiVo[4]. The main concept of CF is very similar to the social recommendation process which is conducted by asking friends with similar tastes. In CF, recommendations are generated based on the usage or rating histories of users and a similarity measure between these histories. For a specific user u_i out of the set of all users U, a set of users with similar histories is identified from the service's larger group of users. In the recommendation step, specific items i_j, typically chosen from the set of all available items I are recommended. Commonly, items included in histories of similar user that have not been used or rated by the current u_i, are considered to be possible recommendations. Accordingly, different recommended items are ranked based on the overall agreement of the similar users in their opinion of them. This agreement is given by user ratings where $r_{i,j}$ describes the rating of user u_i for item i_j. According to [Zho09], CF mechanisms are grouped into the following different classes:

- Heuristic-based: Heuristic-based approaches, which are also called memory-based, calculate ratings based on the foundation of the entire rating set of all users for different items. The utility (usefulness) of an item i_j for a given user u_i is calculated by an aggregation function (cf. section 4.3.2), where the ratings of other users for i_j are aggregated. Additionally, the similarity measure between u_j and the other users is frequently incorporated into the aggregation as a weighting factor for the ratings. Heuristic-based CF approaches are further grouped with the nearest neighbor and the graph-based class. The first class combines the typical nearest neighbor approach where a specific number of neighbors is found by measuring their similarity. Most popular similarity measures are the Pearson correlation, the Cosine similarity and the Euclidean distance (cf. section 4.3.1).

- Graph-based: The graph-based class uses different approaches from the field of graph theory which are commonly applied on graphs where nodes represent users or items,

[1] Amazon – http://www.amazon.com/
[2] Ebay – http://www.ebay.com/
[3] Netflix – http://www.netflix.com/
[4] TiVo – http://www.tivo.com/

and the edges between them indicate the strength of the relation between these nodes.

- Model-based: Model-based approaches use the dataset to train a kind of user-rating model. Afterwards, this model is used to predict missing ratings. For model construction, typical machine learning methods, such as artificial neural networks or Support Vector Machines, probabilistic models such as Bayesian networks or hidden Markov models, or even clustering techniques are used.

Table 4.1 shows an exemplary user rating matrix, where each row represents a user, and each column an item. Each entry of the matrix (e.g. $r_{i,j}$) represents the rating of a user (u_i) for an item (i_j). $\varnothing$ has been used to indicate that a specific item has not been rated by a specific user. In many fields of application, the number of missing ratings is much higher than the number of available user ratings, leading to a typical sparsity problem which is discussed later in this section. As shown in table 4.1, the rating of items can be done with the help of an ordinal scale. Here, the value 1 refers to a a very bad rating, and 5 to a very good rating. In other cases, labels such as ”bad,” ”medium” and ”good” or symbols such as stars or smilies are used. According to [Sch07] other frequently used rating scales are unary or binary. The unary scale offers only one rating option, such as ”good,” which is then either marked by the user, or not. Binary ratings can take both values such as ”thump up” (”good”) and ”thump down” (”bad”). Although, detailed scales offer a way to represent one’s opinion very precisely, prediction accuracy is often controversial (cf. [Cos03]). Considering the user-item matrix, there are two main approaches to the CF process, distinguished by how they incorporate data. These are:

- User-based: To recommend items or predict ratings, user-based approaches consider the similarity between users. This approach’s fundamental assumption is that a user will probably like items that other users with similar interests have liked in the past. It assumes that the user belongs to a group sharing similar interests. In the user-item matrix, each user profile is represented by a row vector of the matrix. Thus, a current user’s row-vector is compared to the row-vectors of all other users to find the most similar ones. To find items of interest for a specific user, the ratings of similar users are analyzed. Items frequently liked by similar users are then recommended, and their ratings are aggregated and used as the predicted rating for the current

	i_1	i_2	i_3	i_4	i_5
u_1	4	∅	4	1	1
u_2	∅	∅	∅	∅	∅
u_3	4	5	∅	2	1
u_4	∅	3	∅	∅	2
u_5	1	2	∅	∅	3
u_6	3	3	3	∅	4
u_7	3	5	∅	2	3

Table 4.1: Exemplary User-Item matrix for 7 users and 5 items.

user. Thus, the whole process relies on the concrete selection of a similarity and an aggregation function (cf. section 4.3.1 and 4.3.2).

- Item-based: In the item-based approach, column vectors, each representing all user ratings of a common item, are used in the CF process. Its fundamental assumption is that a user will probably favor items that are similar to items he liked in the past, as opposed to other items that are not. To predict a rating of a specific item i_j for a user u_i, which has not been rated yet by u_i, the following two steps are conducted: First, the similarity between item i_j and all other items rated by u_i (the set $I^{'} = \{i_x \in I | r_{u_i,i_x} \neq \varnothing \; and \; u_i \in U\}$) is computed. Then, the rating for i_j is predicted by an aggregation function based on the ratings of other users, weighted by their item similarity score to i_j. Generally, this is done for the K most similar items out of $I^{'}$.

Comparing user- and item-based approaches, some important differences are evident. With the user-based approach, the whole dataset must be analyzed. Because user profiles frequently change over time, preliminary offline calculation of profile similarities is not reliable. In contrast, item-based approaches usually compute similarity scores for each item pair offline and keep them in a item similarity matrix (cf. [Mir09]). This can be done because items are more stable than user profiles. Furthermore, for most systems, the item count is much smaller than the count of users. For instance, the dataset used in the Netflix competition, the Netflixprize[1], contains almost 500.000 users compared to only about 18.000 items (movies). Thus, measuring the similarity between items is easier to compute in terms of memory consumption and computational effort. For new users with only few ratings, measuring the similarity to other users is hardly possible. However, with item-based approaches, deriving useful ratings and recommendations, may already be possible. In spite of these differences both methods are very similar on the conceptual level. In both approaches, vectors have to be compared using a similarity measure. Accordingly, the final similarity score is calculated by an aggregation function. The most common similarity measures will be discussed in section 4.3.1 and several aggregation functions in section 4.3.2.

Challenges

Although CF is a well established and well known method in the field of recommendation systems, it suffers from several fundamental problems. According to Schafer et al. [Sch07] and Zhou et a. [Zho09], these problems will be briefly discussed in the following:

- Cold Start Problem: A basic problem of many recommendation mechanisms is the cold start problem. It describes the situation where accurate recommendations can not be generated due to the lack of initial data about users and items. This problem can be further partitioned into the new user and the new item problem.

  The first problem usually occurs when a new user is introduced into the system. Initially, no data about this user, such as ratings or other preferences, is available. Because of this, a group of similar users cannot be identified and subsequently used

[1] Netflixprize – http://netflixprize.com/

in the recommending of items or the prediction of ratings in a personal manner. However, several ways exist to ease this situation. The simplest way is to force users to rate some items or specify their preferences upon registration. Furthermore, non personalized recommendations derived by the population's average ratings can also serve as first "recommendation guesses" for the recommender system. Similarly, basic demographic information such as sex, age or habitation can be used to define a first "demographic" group for the new user.

The second problem, the new item problem, describes the situation when a new item is added to the system. Due to the lack of ratings for this item, it can not be used in the recommendation generation process. This leads to a delicate situation where users usually do not rate the new item because it is not recommended to them, and therefore, is unlikely to be discovered and further recommended. This problem might be addressed by randomly selecting items with few ratings, and asking users to rate them, or by combing CF with other techniques (cf. section 4.4).

- Sparsity Problem: CF makes use of ratings solely for the purpose of calculating similarities among items and users, and generate recommendations. Thus, a very well filled user-item matrix is desirable. In many areas of application, especially in e-commerce, we are facing situations where hundreds of thousands, or even millions of items and users are present in the systems. As users are not able, or in most cases willing to rate thousands of items, this leads to a very low degree of completion and the so called "sparsity problem." In this situation users or items are not comparable anymore, because of a missing common basis of ratings or items. To remedy this issue, many approaches make use of dimensionality reduction and approximation techniques such as Singular Value Decomposition (SVD) [Pol05, Pat07] or restrict their service to more common grouping properties, such as demographic attributes to find similar users.

- Scalability Problem: With the expanding size of a user-item matrix, scalability becomes an issue. The main trouble in such systems are memory consumption and the time the generation of recommendations takes. Depending on the area of application, the time factor as well as the memory consumption may also become critical issues as it is in the realm of TV.

4.3.1 Similarity Measures

Similarity measures model the relationship between two vectors, $\vec{x}$ and $\vec{y}$. Typically, the symmetry $(sim(x,y) = sim(y,x))$ and the identity property $(sim(x,x) = max$, when max is the maximum similarity, which is often 1.0) hold for all similarity measures. For solely positive inputs, the positivity property $(sim(x,y) \geq 0)$ also holds. Nevertheless, in [Mil07] Millan et al. also discuss the application of asymmetric user similarity measures in CF.

In general, all measures mentioned can be used for the user- as well as for the item-based approach by simple replacing item vectors with user vectors in the equations and vice versa. Throughout the similarity measures, $I_{u_x u_y}$ denotes the set of items, which have been corated by user u_x and u_y. This set is defined as follows: $I_{u_x u_y} = \{i \in I | r_{u_x,i} \neq \varnothing \; and \; r_{u_y,i} \neq \varnothing\}$. In the following section, several widely used similarity measures are presented:

- Cosine similarity: The Cosine measure calculates a similarity score by determining the Cosine of the angle between the two vectors. If the input data is positive, the score lies within the interval [0,1] with values near 1.0 denoting high similarity, and values near 0.0, dissimilarity. Aside from CF, this measure is also widely used in the field of text mining and data mining. The Cosine similarity for two vectors $\vec{x}$ and $\vec{y}$ is defined as follows.

$$sim(\vec{x},\vec{y}) = cos(\vec{x},\vec{y}) = \frac{\vec{x} \cdot \vec{y}}{||\vec{x}||||\vec{y}||} = \frac{\sum\limits_{i \in I_{xy}} r_{x,i} r_{y,i}}{\sqrt{\sum\limits_{i \in I_{xy}} r_{x,i}^2}\sqrt{\sum\limits_{i \in I_{xy}} r_{y,i}^2}} \tag{4.1}$$

  $||\vec{x}||$ stands for the Euclidean distance, also called the Euclidean norm of the vector $\vec{x}$ and $\cdot$ denotes the dot-product.

- Tanimoto coefficient: The Tanimoto coefficient calculates the ratio of the intersection and the union of two datasets. It makes use of the Cosine similarity, to extend the so called Jaccard index. Because of this, it is often referred to as the extended Jaccard index and formulated as follows:

$$sim(\vec{x},\vec{y}) = \frac{||\vec{x} \cdot \vec{y}||^2}{||\vec{x}||^2 + ||\vec{y}||^2 - ||\vec{x} \cdot \vec{y}||^2} = \frac{\sum\limits_{i \in I_{xy}} r_{x,i} r_{y,i}}{\sum\limits_{i \in I_{xy}} r_{x,i}^2 + \sum\limits_{i \in I_{xy}} r_{y,i}^2 - \sum\limits_{i \in I_{xy}} r_{x,i} r_{y,i}} \tag{4.2}$$

- Euclidean distance: The Euclidean distance is defined by the distance of a straight line between two data points in vector space. It is one of the most common distance measures and often simply referred to as ”distance.” In the n-dimensional space it is defined as follows:

$$euclid(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4.3}$$

  As a distance function, it provides, with rising similarity, smaller values. To reflect the common comprehension of similarity, it is modified as follows:

$$sim(\vec{x},\vec{y}) = \frac{1}{1 + \sqrt{\sum\limits_{i \in I_{xy}} (r_{x,i} - r_{y,i})^2}} \tag{4.4}$$

  Similarity values vary between 1.0 for identical vectors and 0.0 for completely dissimilar vectors.

- Pearson correlation: A more advanced similarity measure is the Pearson correlation coefficient. It measures the degree of linear relationship between two datasets. Compared to the Cosine measure it also incorporates the average ratings of users. Thus, it is able to correct for grade inflation. Grade inflation emerges when a particular user tends to give constantly higher ratings than others as a result of different perceptions of the rating scale [Seg07]. This correlation is formulated as follows:

$$sim(\vec{x},\vec{y}) = \frac{\sum\limits_{i \in I_{xy}} (r_{x,i} - \overline{r}_x)(r_{y,i} - \overline{r}_y)}{\sqrt{\sum\limits_{i \in I_{xy}} (r_{x,i} - \overline{r}_x)^2 \sum\limits_{i \in I_{xy}} (r_{y,i} - \overline{r}_y)^2}} \tag{4.5}$$

In this equation $\overline{r}_x$ denotes the average rating of user u_x.

- Spearman's Rank Correlation: The Spearman's rank correlation coefficient defines a special version of the Pearson correlation coefficient. It can be understood as the Pearson correlation coefficient calculated on the rank of each variable. In contrast to the Pearson correlation, it is focused on ordinary variables (the rank) and does not assume a linear relationship between the variables. It is formulated as follows:

$$sim(\vec{x},\vec{y}) = \frac{\sum_{i=1}^{n}(rg(x_i) - \overline{rg}_x)(rg(y_i) - \overline{rg}_y)}{\sqrt{\sum_{i=1}^{n}(rg(x_i) - \overline{rg}_x)^2}\sqrt{\sum_{i=1}^{n}(rg(y_i) - \overline{rg}_y)^2}} \tag{4.6}$$

As a preliminary step to the similarity calculation, the rank of each dataset (e.g. $rg(x)$) is determined. This is done by ascendantly ordering the elements (e.g. x_i) of each vector. The rank is defined by the vector element's position in the ordered list. $\overline{rg}_x$ denotes the average rank of x. Compared to the Pearson correlation, the Spearman coefficient is more resistant to outliers.

4.3.2 Aggregation Functions

After calculating the similarities between items or users, the aggregation step combines these scores and ratings into an overall prediction. Again, the aggregation can be done for both the user- as well as for the item-based approach. The aggregation can be done in various ways, however the following two aggregation functions are frequently used:

- Basic Aggregation: The easiest way to get an overall prediction is to calculate an average. Determining a specific rating $r_{i,j}$ for user i and item j can be done by simply analyzing other user's ratings for item j. For users, this can be formulated as follows:

$$avg(U^{'}) = \frac{1}{|U'|} \sum_{u \in U'} r_{u,i_j} \tag{4.7}$$

where $U^{'} = \{u \in U | r_{u,j} \neq \varnothing \; and \; u \neq u_i\}$.

- Weighted Aggregation: The weighted aggregation incorporates the similarity scores into the aggregation step. Thus, more closely related items or users contribute more to the average than others. The weighted aggregation is formulated as follows:

$$wavg(U^{'}) = \frac{1}{\sum\limits_{u \in U'} sim(u,u_i)} \sum_{u \in U'} sim(u,u_i) r_{u,i_j} \tag{4.8}$$

4.4 Hybrid Approaches

Recommendation systems suffer from different incapabilities. Content-based approaches perform quite well in personalized scenarios where adequate information about items is available, however they also tend to overspecialization. On the other hand, collaborative approaches are able to provide diverse and unexpected recommendations, but suffer significantly from cold start and sparsity problems. Thus, hybrid recommendation approaches try to combine two or more recommendation approaches in order to overcome, or at least ease each's weaknesses. There are several ways to combine approaches into a single system. In the following, we will discuss the major approaches according to [Bur02, Ado05]. In general, these combinations make use of a collaborative filtering component and extend it by other recommendation approaches. Often, a content-based approach is used as an extension. In the following section, three approaches will be discussed, in which the recommendation components are completely independent of each other. This means that the scores of the different recommender components are kept separate. These elements are as follows:

- Switching: The main assumption of the switching combination is that different recommendation approaches perform better than others in different circumstances. Thus, a system employing a content-based and a collaborative approach may decide to use the collaborative mechanism in situations where no appropriate information about an item is available, but may switch to the content-based approach if an item has only rarely been rated.

- Weighted: Different recommendation approaches are combined in form of a linear-weighted combination. The score of each recommender is weighted and summed up to provide an overall score for a specific item. Depending on the accuracy of each recommender, the weights may be adjusted over time.

- Mixed: In situations where a large number of recommendations is desirable, a mixed combination may be used. It combines recommendations generated by different methods into one comprehensive view. Each component separately contributes with its recommendations to this view. This combination method is said to avoid the new item, as well as the overspecialization problem.

In contrast to hybrid approaches like switching, weighted and mixed, the following approaches combine different recommendation methods in a fixed order:

- Cascade: In the cascade method, different recommendation mechanisms are combined in a stepwise manner. The first mechanism produces a rough set of recommendations, whereas the next mechanisms are used to refine them in cases of poorly or very similar ratings. Thus, further process steps are only conducted if the recommendations of the previous step are not ”adequate” enough.

- Feature Combination: This approach combines the recommendation approaches by using features of one method as additional information for the other method. For instance, this combination may be done by extending typical content information from the content-based approach by using the features of a collaborative approach such as user ratings. Finally, based on this extended data basis, the content-based approach is used to generate the recommendations. For sure, the integration of content-based information into a collaborative recommendation step would also be possible.

- Feature Augmentation: This combination is based on feature augmentation, and uses one recommendation mechanism to produce an output that serves as an expansion of the input for the next component. For instance, a content-based approach may be used to generate ratings that are then used in combination with the users' ratings in a collaborative step. This helps in overcoming the new item problem. Conversely, the output of a collaborative component could also be used to ease the overspecialization of a content-based filtering step.
- Meta-level: Compared to the feature augmentation approach, the meta-level combination uses the entire model generated by one recommendation mechanism as the input for the next mechanisms. As an example, a content-based approach may be used to generate a feature vector that serves solely as an input to a collaborative filtering step. The recommendation generation relies completely on the input generated by the previous component.

Even though the combination approaches presented are able to cope with some of the problems of recommendation systems, the cold start problem still remains. All mechanisms need a set of ratings to function properly. Nevertheless, several hybrid recommendation systems and studies such as those found in [Bur07], [Spi09] and [Gha10] have demonstrated their superior performance and a lot of synergies compared to the use of one single recommendation mechanisms.

CHAPTER 5

Session Mobility and TV Add-On Services

In the discussion of user-centrism in multimedia, session mobility and additional multimedia services are important topics. In the following chapter, we will focus on the realm of TV and multimedia consumption, in order to provide feasible approaches to both topics.

The main aim of session mobility is to help people cope with the increasing number of different devices in use. Although manufacturers are attempting to build a single device that is suitable for most tasks, a lot of people use devices designed to complete specific tasks. With the increasing variety and uses of different devices, more and more users are finding themselves in the unpleasant situation of having their digital life spread over several devices. As this trend continues, it becomes a nuisance trying to keep the appropriate data available to the right devices. Because of this, there is a need for an automated system that can continue the work currently being done by other devices, without wasting too much time. Session mobility enables the user to disconnect a session from one device and transfer it to another. Through the use of an intermediary migration step, the user is able to use different services on different devices in an seamless way.

In the realm of TV, digitalization has played a critical role in creating new opportunities for enhancing the TV-watching experience. Much work has been done to provide additional multimedia services to accompany traditional TV services. Although currently very limited, the Electronic Program Guide (EPG) gives a first impression of the possibilities of future TV services. Unfortunately user-centrism is not often taken into consideration during the development of new services and technologies. Important for the concept of user-centrism (cf. section 1.1) is that devices should provide users with an easy way to access services and applications. Thus, the current challenge is to combine mobility, additional services and easy access into a single approach, to bring user-centrism into TV related services.

This chapter is structured as follows: In section 5.1 we introduce our approach, called Agent based Session Mobility (AbSM), which aims to provide session mobility on different devices. Based upon an agent platform most tasks such as the migration process or providing services, are handled by build-in functions of the platform. Section 5.2 provides details about our approach to enhance the traditional TV watching experience by introducing add-on services to mobile devices. Using an ad-hoc service architecture, the principle of easy access is also taken into account.

5.1 Agent based Session Mobility (AbSM)

Our Agent based Session Mobility component has been designed to meet demands for session mobility in mobile environments. The user can access his or her data on any physical host that is part of the AbSM platform. The system utilizes the technology of mobile agents to meet the user's need for mobility. With such a component, the user does not need to worry about the location of the current session of his or her work. For reasons of compatibility and in order to provide a standardized format - MPEG-21 - has been used to represent the actual session.

The following usage examples illustrate the benefits of AbSM:

- ”Personalization” - mobility: The introduction of session mobility makes it possible to enhance and improve the way content is selected. A user profile and a filtering engine contained within the session can be used to personalize access to different types of content and perform user preference based filtering in various situations, regardless of which device is currently in use. The profile could also be enriched and adapted in a transparent manner, based on the use of this filtering component and the context in which it is used (e.g at home, in the office, etc.).
- ”Browsing sessions” - mobility: Another example of the use of mobility within this system is that of ”browsing sessions.” The simplest form of a ”browsing session” could be made up of a collection of websites and their associated metadata (e.g. their access dates or cookies). By enabling mobility for this kind of sessions a user would be able to transfer his active session from his PC to his smartphone without further effort and help of a central component such as a server. As a result, the user is able to continue surfing the web using his smartphone as he goes on his way.
- ”Video sessions” - mobility: Consider this scenario: someone is sitting in the living room in front of his TV-set and watches football. Because it seems like nothing important is happening, that person decides to watch the game with a good friend two blocks away. After leaving his house, he notices a multitude of cheers from his neighbors. When he arrives at his friend's house, he has missed the decisive goal. With the use of AbSM, that person could have continued watching, or at least listening to, the broadcast on his PDA or smartphone by transferring the live session from his set-top box to his mobile device. By tracking the location of the user, the session migration to the mobile device could even be done automatically when the user is on the move.

The rest of this section is organized as follows. Sections 5.1.1 and 5.1.2 give background information on the MPEG-21 standard and on mobile agent systems. A detailed discussion of the Java Agent DEvelopment framework (JADE) and its benefits to our system is shown in section 5.1.2. Sections 5.1.3 and 5.1.4 present our system in detail. Aside from the architecture, the selection of the target hosts, based on basic benchmark results, and the session migration will be discussed as well. To evaluate the applicability of AbSM section 5.1.5 presents our demonstration platform. In section 5.1.6, we take a look at several similar projects and discuss the main differences, as compared to our proposal. Finally, section 5.1.7 concludes the description of AbSM with a short summary and future work.

5.1.1 MPEG-21

A large number of different frameworks for the consumption and delivery of multimedia content are available. Most of them cover only basic functional and organizational components. In contrast to these approaches, MPEG-21 tries to create a standardized ”big picture” for multimedia systems. It puts into place a framework to act as a shared basis for the ”seamless and universal delivery of multimedia” [Bur06]. MPEG-21 strives to guarantee a transparent use of multimedia resources across different networks, devices, user preferences and communities for all players in the delivery and consumption chain. The standard consists of 18 parts which are centered around 7 key areas:

1. Digital Item Declaration
2. Digital Item Identification and Description
3. Content Handling and Usage
4. Intellectual Property Management and Protection
5. Terminals and Networks
6. Content Representation
7. Event Reporting

MPEG-21 introduces two key concepts: the Digital Item (DI) and the User. The DI (the ”what”) is made up of the content, along with its resources, metadata and structure. The user (the ”who”) is defined as any element that interacts with the MPEG-21 framework or uses a DI. For session declaration and mobility, as is the focus of AbSM, only the areas of *Terminals and Networks* and *Digital Item Declaration* are of further relevance. Nevertheless, other applications providing presentation or playback functions must also cover most of the other MPEG-21 key areas as discussed in [Ran08]. For a comprehensive overview of the MPEG-21 standard, the reader can refer to [Bur06].

Content DI and Context DI

In describing the content of its associated session, AbSM makes use of two MPEG-21 concepts: the Content Digital Item (Content DI) and the Context Digital Item (Context DI)[Int05]. Both of them are described in the form of XML files that use the MPEG-21 Schema definitions. The Content DI represents the actual content containing resources, the metadata and their interrelationships. It may also offer several choices for specific content, each providing a different level of quality, format, media type and resolution. Because of this, Content DI can be processed, based on these choices, in a variety of ways, each adapted for a different device. For example, a Content DI may contain a movie in 3 selectable qualities and a transcript of the movie for devices without video playback capabilities.

The Context DI saves the actual session information. It contains information about the playback conditions, e.g. what choices were made to play back the Content DI and the actual playback time. The session is saved in a Digital Item Adaption Description Unit, which is a type of *SessionMobilityAppInfo*. This type of description is used to adapt the Content DI. It also stores information about the application that uses the content. Because the information used to describe and retain a session varies from application to application, the format of the session description has not been standardized. [Bur05]

Terminal Capabilities

In order to describe the capabilities of different devices, AbSM adopts a mechanism for characterizing terminals from MPEG-21. All relevant properties of the hardware are saved in a MPEG-21 conformant format. These capabilities can be classified into three categories.

1. Device capabilities: This category includes attributes like DeviceBenchmark, where a measure for CPU performance can be saved, StorageCharacteristics where values such as the size of main memory can be noted, the DeviceClass (PC, laptop, PDA, etc.) and PowerCharacteristics, like the remaining battery time.

2. Codec capabilities: CodecCapabilities define the capabilities of devices to encode and decode audio and video.

3. Input–output characteristics: The I/O category includes information about Human Interface Device (HID) capabilities, visual output like display resolution and color-depth, and audio output specified by number of channels, bits per sample, sampling frequency etc.

A fragment of a typical Terminal Capabilities description is shown in listing 5.1. By examining the elements of this description, several different conclusions about the selection of a device for transferring a current session may be drawn. For instance, because of the information described in the *PowerCharacteristics* element, a slower device with a higher remaining battery time may be favored over a faster one with a very low battery. In order to choose the way a DI is processed e.g. download, progressive download or streaming, *StorageCharacteristics* must be considered. Elements of the I/O and of the *Codec capabilities* have the greatest impact on the ability of a device to process a DI and on how it is presented.

Listing 5.1: Excerpt of a MPEG-21 Terminal Capabilities description.

```
<TerminalCapability xsi:type="PowerCharacteristicsType"
 batteryTimeRemaining="4200" />#
<TerminalCapability xsi:type="StoragesType">
 <Storage>
  <StorageCharacteristic
   xsi:type="StorageCharacteristicsType"
   size="5177344" />
 </Storage>
</TerminalCapability>
<TerminalCapability xsi:type="DeviceClassType">
 <DeviceClass
  href="urn:mpeg:mpeg21:2003:01-DIA-DeviceClassCS-NS:1">
  <mpeg7:Name xml:lang="en" />
 </DeviceClass>
</TerminalCapability>
```

```
<TerminalCapability xsi:type="DisplaysType">
 <Display>
  <DisplayCapability xsi:type="DisplayCapabilityType"
    colorCapable="true">
   <Mode>
    <Resolution horizontal="1920" vertical="1200" />
   </Mode>
   <ColorBitDepth blue="32" green="32" red="32" />
  </DisplayCapability>
 </Display>
</TerminalCapability>
<TerminalCapability xsi:type="AudioOutputsType">
 <AudioOutput>
  <AudioOutputCapability
    xsi:type="AudioOutputCapabilitiesType"
    numChannels="2">
   <Mode bitsPerSample="16" samplingFrequency="44100" />
  </AudioOutputCapability>
 </AudioOutput>
</TerminalCapability>
 <Decoding xsi:type="VideoCapabilitiesType">
  <Format href="urn:mpeg:mpeg7:cs:VisualFileFormatCS:2001:8">
   <mpeg7:Name xml:lang="en">
    H263
   </mpeg7:Name>
  </Format>
 </Decoding>
</TerminalCapability>
```

5.1.2 Agent Systems

A software agent is generally a label for any kind of program that carries out tasks and makes decisions on behalf of a user or another program. One example is a search bot that scours the web for useful information. It decides whether or not a website has relevant information and extracts useful knowledge for the user. Such agents are often defined by their characteristics, namely autonomy, proactivity, reactivity, adaptivity, persistence and social-ability. Autonomy means that the agent can act on its own, without the interaction of the user. Proactivity means that it can act of its own will. The agent may, however, still need to ask for permission before doing anything that might be potentially harmful. When an agent can react to its environment and adapt to changes in it, it can be viewed as displaying reactivity. In order to accomplish this, it may need to gather information about its surroundings. Persistence means that agents continuously run, and are not stopped after their task has been finished. The ability of agents to communicate and cooperate with other agents and system components is called social ability. In addition to these characteristics, mobile agents also have the ability to move from a current, to another host. This action is called ”migration.” For a detailed explanation of agent systems, see [Pad05, Bag07].

Several agent systems have been evaluated for their application in our AbSM component. Driven by the need for mobility and the support of different platforms and devices e.g.

handhelds and smartphones, we have focused on Java based platforms. These include Aglets, Beegent, Bond, JACK, MIA, UbiMAS, Mole, Voyager, Grasshoper, Gypsy, Cougaar, Agent Factory and JADE. Most of these agent systems were discarded for being out-of-date, and for their failure to support mobile environments. The most promising platforms were JADE and Agent Factory. Finally, JADE was chosen for its mature status and its broad support of different Java Runtime Environments, like J2ME in both configurations - the Connected Device Configuration (CDC) and the Connected Limited Device Configuration (CLDC), Personal Java and J2SE. For a comparative discussion of several mobile agent platforms, interested readers should refer to [Tri07].

JADE

Java Agent DEvelopment (JADE) framework [Bel07] was developed and distributed by Telecom Italia, but published under the Lesser General Public License Version 2 (LGPL2). It is a Java based agent framework which fully complies with FIPA[1] standards [Fou09].

A JADE platform provides the physical infrastructure for agents that may spread over several physical hosts. It consists of several agent containers which can be kept on different hosts. Each one is a multithreaded runtime environment for JADE agents. One agent container always acts as the main container, which hosts organizational information and processes for the whole agent platform. All standard containers have to register at a main container. At any given point in time, there should only be one active main container per agent platform. The organizational information and processes hosted by the main container are as follows (see figure 5.1):

- The Container Table (CT), acts as a registry for all other containers, storing their transport addresses and object references.

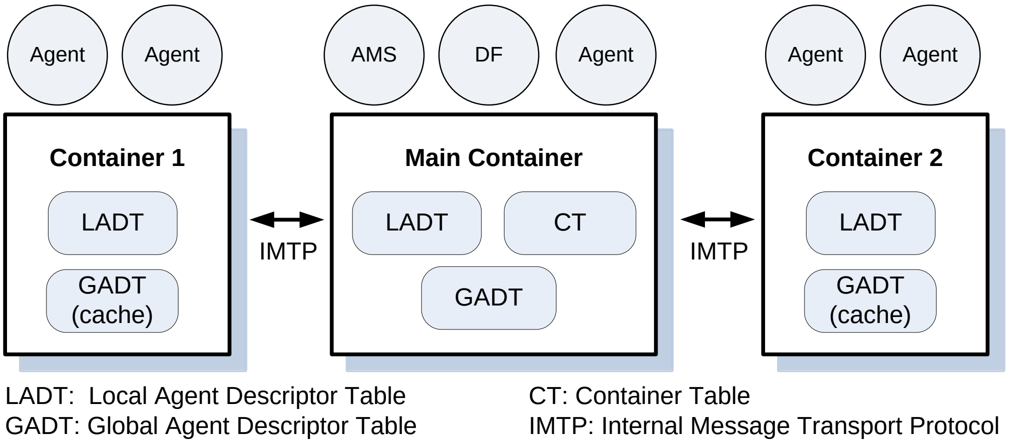


Figure 5.1: Organizational structure of the Java Agent DEvelopment (JADE) framework.

[1] The Foundation for Intelligent Physical Agents provides a collection of standards to ensure interoperability between different FIPA compliant agent systems.

- The Global Agent Descriptor Table (GADT) holds information from all registered agents.
- The Agent Management System (AMS) agent features most organizational tasks like registering new agents, de-registering agents upon deletion, and taking care of the whole migration process.
- The Directory Facilitator (DF) agent provides a registry of services offered by different agents. This mechanism is similar to the yellow pages of the Universal Description, Discovery and Integration[1] (UDDI) service, used in the Web Service technology.

The main container is the central component of the platform. It hosts the most important agents, the Directory Facilitator (DF) and the Agent Management System (AMS) agent, which are only present in the main container. Nevertheless, most operations do not need the main container at all, as every container keeps its own copy of the GADT. This copy is called the Local Agent Descriptor Table (LADT). If the local table is out of sync, the container triggers the refreshment of its cache. Typically this happens when a queried entry, such as the address of a specific agent, can not be resolved by the local table, or if a querying agent reports that the information was inaccurate. Because of the agent cache, the main container is only involved when agents or services are created, deleted or changed (which includes agent migration). However, the main container is still a single point of failure, considering a crash can disable the agent platform. The Main Container Replication System (MCRS) helps to overcome this situation by starting several ”sleeping” main containers, which simply act as a kind of proxy to the active main container. In the event that the current main container fails, the remaining main containers are notified, and can reorganize accordingly.

Message transport is implemented in two different ways in JADE: For inter-platform communication, an HTTP interface is available. This is the entry point for messages sent from outside into the platform - a point that complies with the FIPA standards. For intra-platform communication, a special protocol called Internal Message Transport Protocol (IMTP) is used, which does not comply with FIPA. Because the IMTP is solely used within JADE, it is tailored to JADE’s needs and therefore more efficient and better suited. Besides standard message exchanges between agents, it is also used for system-message exchange. One example of this is the command to shutdown a container or to kill an agent.

JADE-LEAP

The limitations of the hardware of mobile devices (J2ME CLDC) made it necessary to introduce a special Lightweight Extensible Agent Platform (LEAP) for JADE. LEAP uses the so-called split-container mode, where a lightweight front-end container is used on the mobile device, which keeps a constant connection to a back-end container on a J2SE host. Neither the front-end, nor the back-end are a container on their own, because both only provide parts of the functionality - hence the name ”split-container.” Because the serializing and deserializing of objects is not supported by J2ME CLDC, ”real” migration is

[1] http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm

not possible. Thus, commonly migration is realized only in cooperation with the back-end container.

Unfortunately, this also limits the abilities of the agents used in AbSM. AbSM agents can only achieve mobility by one of two distinct means: Either they are "mobile agents" as in "running on mobile devices" (using Jade-Leap and no direct migration), or they are "mobile agents" as per definition, able to migrate, but unable to be directly used on mobile devices.

Benefits of Agent based Systems

Our platform strongly benefits from the employment of an agent system. The main advantages are as follows:

- **Platform independency**: Due to the use of JADE, our system can be used on all platforms that support JADE. Based on Java, interoperability of routines and methods embedded in agents, is guaranteed as well.
- **Management of distributed components**: Most organizational tasks for offering, finding and migrating to a target host are provided by the agent system (see CT, GADT, AMS, etc.).
- **Mobility**: The transportation of the session information is fully handled by the agent system.
- **Expandability**: To cope with future demands and developments, the system can be easily extended using new session types and target selection mechanisms (cf. section 5.1.3), because they are fully embedded in the agents. Thus, a new method must only be integrated in the agents in order to extend the functionality of the whole system. JADE, as an agent platform that conforms to FIPA, also enables interoperability and a comfortable integration with other agent platforms that likewise conform to FIPA.

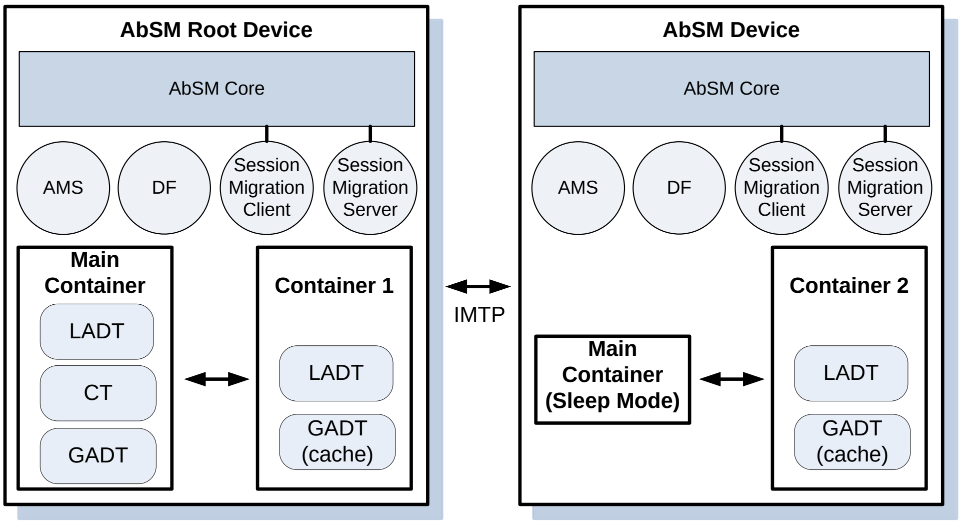


Figure 5.2: Overview of the Agent based Session Mobility (AbSM) architecture.

- **Reliability**: In our platform, an agent migrates directly to the target device, where the session is continued autonomously. Thus, no connections to the originating host or a server need to be maintained, and work could also be continued in situations where connection problems are likely to lead to communication interrupts and failures.

5.1.3 Description of the System

AbSM has been implemented in Java, primarily to achieve platform independence and as a way to provide support to a broad range of mobile devices. Figure 5.2 shows a rough overview of an AbSM device and its relationship to the services provided by the agent platform. JADE is used in the system for message delivery and to handle the entire migration process. Each device in the AbSM platform contains a *Session Migration Server* agent, a *Session Migration Client* agent and several JADE specific components such as DF, the AMS and different tables holding organizational information. The *Session Migration Server* agent is used to offer the current device, through the use of a migration service, as a potential migration target. Additionally, the *Session Migration Client* agent makes possible the discovery of migration targets and helps with the task of migration. To enhance the fault tolerance, each instance of AbSM which uses J2SE has its own agent container, as well as its own main container. However, at any given time, only one of the networked devices' main containers is active, the others are working as proxies. Based on the MCRS (cf. section 5.1.2), the main containers of different devices are organized in a ring, where every local main container maintains a connection to the next main container. If the connection to the active main container is lost, the ring will re-organize and one of the other main containers changes its status to active. This mechanism guarantees a maximum degree of fault tolerance. Without it, the whole system would collapse instantly if the JADE MainContainer was to crash, due to the loss of most organizational information of the agent system.

Upon startup, each new device must register its MigrationServer agent at the Directory Facilitator of the agent platform. For this reason, the first AbSM device creates a main container and all of its associated "service"-agents such as the DF and the AMS. This device now runs and manages the active main container. To avoid the creation of other active main containers, which would lead to multiple isolated AbSM platforms, each device must send a multicast request in order to discover if other active main containers are present. If the request is received by another instance of AbSM, it will reply, and thus allow the new device to join the existing platform and its MCRS. In a final step, the new device starts and registers itself with the MigrationServer agent. This agent manages all incoming migration requests from other devices and negotiates the session handover. The MigrationClient agent is only activated if an actual session should be moved to another device. A detailed description of the migration process can be found in section 5.1.4. The communication between AbSM Devices is handled by JADE, based on the IMTP.

As depicted in figure 5.3, AbSM is made up of several main components that will be described in the following section:

- AbSM Core: This part provides the main functionality of the system. It provides and enables the evaluation of the device's capabilities, migration target selection, the agent code (MigrationClient agent and MigrationServer agent) and the session migration process itself (see section 5.1.4).

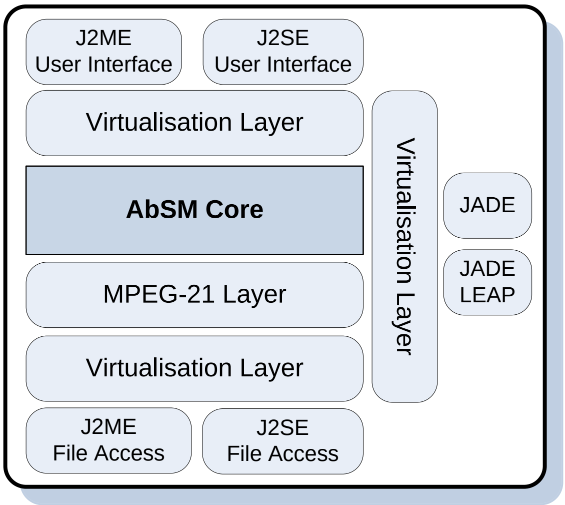


Figure 5.3: AbSM's device structure.

- MPEG-21 Layer: All the information about content, context (what and where the content is) and devices (e.g. benchmark results) are saved in a MPEG-21 conformant XML file. Because of this, the MPEG-21 Layer was introduced to support the creation and processing of MPEG-21 compliant XML documents. A DOM parser was used to allow random access to XML elements. Additionally, this layer adopts the Least Recently Used (LRU) strategy for buffering frequently accessed DI, making access to these items more efficient.

- Virtualisation Layer: J2SE and J2ME share many source code fragments. However, there are major differences in the way user-interfaces are realized, files are accessed or which libraries are supported. Thus, several functions of AbSM must be made available in two versions - one for J2ME and one for J2SE. These functions include the determination of terminal capabilities (eg. supported codecs or resolution and color-depth of the screen), and the storage of digital items. For these reasons, a virtualisation layer has been created. It allows the system to keep as many functions as possible working in both versions. Basically, the classes can be compiled for both versions, while still being able to use edition-specific functions through the virtualisation layer. Thus, most source code fragments can be shared between both editions, with the exceptions of the user interface and some of the elements of the virtualisation layer. The access to the agent platform is managed by this layer as well, in order to keep version specifics out of AbSM's core.

- J2ME/J2SE specific elements: These parts hold all version-specific elements of AbSM such as user interfaces, file access or agent platform specifics and libraries (see section 5.1.2).

Benchmarking and Hardware Evaluation

In order to be able to decide whether or not a device is able to continue a specific session, the hardware capabilities of the machine and a rudimentary scale for comparison has been created. Please note that this measure is not to be confused with a real benchmark. Three main categories were measured on a scale between 0 and 100 points: CPU speed, resolution and color capabilities.

Additionally, as a session specific category, the multimedia capabilities of the devices were evaluated. Thus, for most devices a list of supported audio and video codecs is available.

CPU speed: The CPU speed was measured by calculating a total of 1 million additions, multiplications and divisions. This will take a couple of seconds on a reasonably fast mobile phone (e.g. Nokia N80), and is very fast (split-second) on a personal computer. The time taken for this test denoted by x is measured in milliseconds, with a maximum of 20 seconds for the test.

For slow devices, a good resolution can be achieved by simply using a linear function: For every 0.2 seconds needed to complete the test, 1 point was subtracted from a starting value of 100 points as:

$$f_2(x) = 100 - x \tag{5.1}$$

Unfortunately this means that fast devices that need less than 1 second for the test (which should be most personal computers) will all receive nearly the same rating.

For fast devices a function is used which is based on indirect proportionality between time needed for the test and points, as follows:

$$f_1(x) = \frac{100}{x} \tag{5.2}$$

This results in a high resolution for fast devices, but a much worse resolution for slower devices because devices requiring 10.2 to 20 seconds would all receive 1 point.

By taking the average of both functions (f1 and f2), a satisfactory resolution was achieved for both fast and slow devices (f3). f_3 is defined as follows:

$$f_3(x) = \frac{f_1(x) + f_2(x)}{2} \tag{5.3}$$

Although it is a rather rough measurement, it has proven a useful way to distinguish between slow and very fast devices.

The three functions can be seen in figure 5.4. A logarithmic scale might have performed similarly, but J2ME CDC 1.0 does not support logarithmic or exponential functions.

Resolution: Because displays are 2-dimensional, the square root of the number of pixels was used as a basis for attributing points. A normalizing factor was introduced to grant 100 points to all displays exceeding 2 mega pixels (2000*1000), while displays with 200 pixels (20*10) would still achieve one point.

Color depth: Because the number of colors is usually dependent on the number of bits used to identify a specific color, this value is used for calculating the points. This

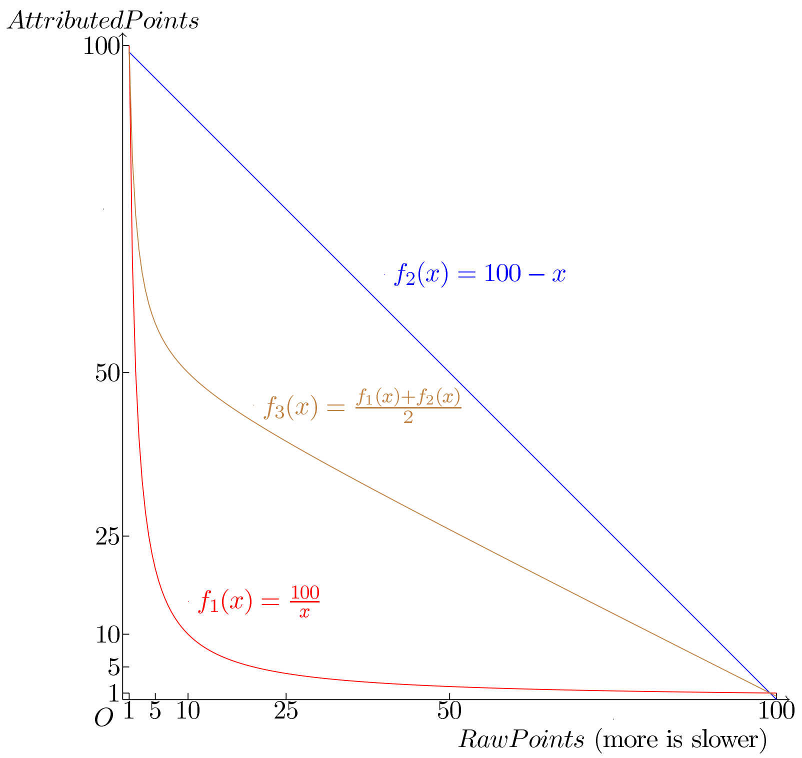


Figure 5.4: CPU raw points - attributed points diagram

allows for the comparison of 8-bit colors with 32-bit colors. Since 32-bit is the current standard for desktops, a normalizing factor was introduced to give a result of 100 points if the color depth was 32-bit. Thus, points are calculated as:

$$\frac{100 * colordepth}{32}$$

To prevent gray scale displays with a high number of gray scale colors from getting a better result than a simple color display, a punishing factor of 5 was used. If a display is only gray scale, it gets only 1/5 of the points that a color display with an equal number of colors would get.

Priorities and Decision: Thanks to the pre-processing described above, there are 3 point-values each ranging from 1 to 100 points that roughly describe the underlying hardware. For a simple result, the 3 where combined to a single decisive value. Weighting factors were also introduced for this step. These factors should generally depend on the application used, in order to reflect the importance of each specific value for the application. For example, colors are important for pictures, while screen size and, as a result, resolution are important for large datasets like tables. Similarly, for some tasks like decryption CPU power is the most important factor. For playback of videos, all three values are important - but only up to a certain point. If the codec is well supported and the CPU is fast enough to decode the video in real-time, more CPU power will not yield further improvements. On the other hand, if the computing power is too low, the video will be choppy.

Thus, a single value trying represent the ability of the underlying hardware to perform a specific task, like playing a video, starting a game or the like, is introduced. By designing special priority classes for each action, a prioritized points value is calculated. Priorities basically assign an importance-value to each of the three main categories that were measured. The idea behind those priority-values is that they shall specify the minimum number of points needed to deliver the best possible service quality. If there is still an improvement from 49 to 50, but not from 50 to 51, then the priority should be 50.

To calculate a single value using the three categories' values, each of them is divided by the priority value that was assigned to that category. That way, a fulfillment ratio for each of the three main factors is created. The final value is then calculated using the bottleneck principle, since it does not matter how fast the CPU is and how brilliant the colors are, if the screen is too small to recognize anything. Thus, the smallest of these three ratios multiplied by 100 determines the final point value. If a category has an importance of 0 (no importance), this category would simply be ignored. If all three categories are attributed with an importance of 0, every system will get 100 points.

Furthermore, the result of these calculations is used for automatic decision making in the migration target selection step. In order to enable this automatism, each Content Digital Item is assigned one point value (and a corresponding priority) per component. This value is then compared with the points that were calculated during the benchmarking and hardware evaluation.

5.1.4 Session Migration

As mentioned before, the session information itself is saved and transferred as a DI in MPEG-21 format. There are several agents involved in the migration process. Figure 5.5 shows the participants and the migration process in a simplified form. Since it does not support serialization and reflection, the migration is not fully supported on J2ME. Because of this, there are two versions for client- and server applications. Depending on whether the actual device is the initiator or the target of the migration process, the corresponding application must be utilized - MigrationClient for migrating to another place, and the MigrationServer to accept and process a migration request. Both components create a corresponding agent on startup. In order to provide seamless migration, the MigrationServer is typically started in the background, waiting for migration requests. When both, the client and the server are running, the migration process can be initiated. If no benchmark and capability information is present upon the startup of the MigrationServer, a benchmark calculation is conducted. The default mean of communication within AbSM is the JADE IMTP system, which is used by all agents. A typical migration process consists of the following 7 steps (see figure 5.5):

Step 1 To become available on the platform for migration, requests from the MigrationServer have to be registered at the DF.

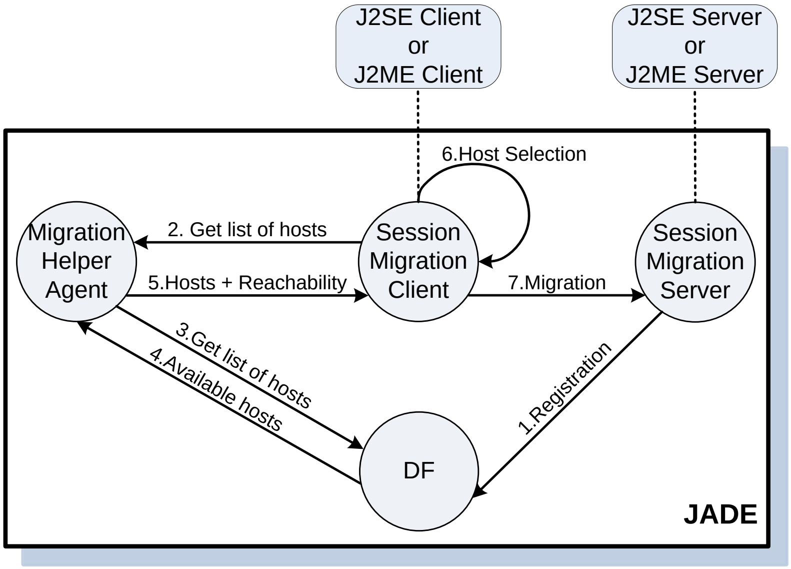


Figure 5.5: Schematic visualization of the session migration process.

Step 2 In order to discover all available migration targets, the MigrationClient agent queries the Migration Helper agent.

Step 3 and 4 The Migration Helper Agent queries the Directory Facilitator (DF) in order to get a list of all available hosts. Then a bluetooth scan is performed to find all available devices and save their bluetooth identifiers. By comparing the bluetooth IDs of the available hosts with the bluetooth IDs of the list of hosts returned by the DF, all hosts are marked as either reachable (nearby) or unreachable (further away). The bluetooth interface is only used to gain a rough estimate on the distance to a possible target host and to provide contact details for them. In general, other, more precise mechanisms for location estimation can also be easily integrated and used in this step.

Step 5 In step 5, the list of possible migration targets (hosts), complete with contact information, is returned to the MigrationClient agent.

Step 6 The host selection can be done automatically or manually. In both variants, the benchmark results and the capabilities of each available host are taken into account, in order to identify the best suited host for the current session. Commonly, different session types lead to different decisions, as they provide different priority factors (cf. section 5.1.3). For instance, automatic migration of a video session will simply pick the host with the highest number of video points, and will initiate migration. In the manual mode, the user is presented with a complete list of all possible migration targets and information about the number of points they achieved. Based on this information, the user then makes the choice of which one to use. Additionally, a user can see whether or not a specific target is within bluetooth range, so decisions can be made based on locality. To be able to distinguish between the different hosts, their hostname is used for J2SE hosts, and the telephone number for J2ME hosts.

In addition to the current session, other digital items available on the client device can be marked for migration as well.

Step 7 Due to the restriction of JADE-LEAP, "real" migration is only supported between J2SE hosts. Thus, our solution to provide session migration on J2ME is to send working instructions specifying the task and the session data to a specialized agent on the J2SE host's side. This agent must then provide the implementation for these instructions, otherwise the migration fails. After the migration is initiated, the digital items are sent to the new host. When all of the digital items have been transmitted, a message is sent to indicate that the migration is complete. During a migration, all migration related messages from other hosts are discarded.

As soon as a migration is finished the MigrationServer initiates and continues the specified session on the new host.

5.1.5 Evaluation - Video Session Scenario

The system was tested with a video-session scenario where two smartphones (a Nokia N80 and a Nokia N95) and a standard laptop were used as shown in figure 5.6. All devices were equipped with a bluetooth interface and connected to a Wireless Network. For video streaming, the Darwin Streaming Server[1] was used. A video was transcoded as a 3gp file for the cell phone in a low quality, and in a higher quality for standard PCs. The 3gp version of the video was started on the cell phone (Nokia N95), paused and the session migration process was initiated. Following the steps described in section 5.1.4, the session

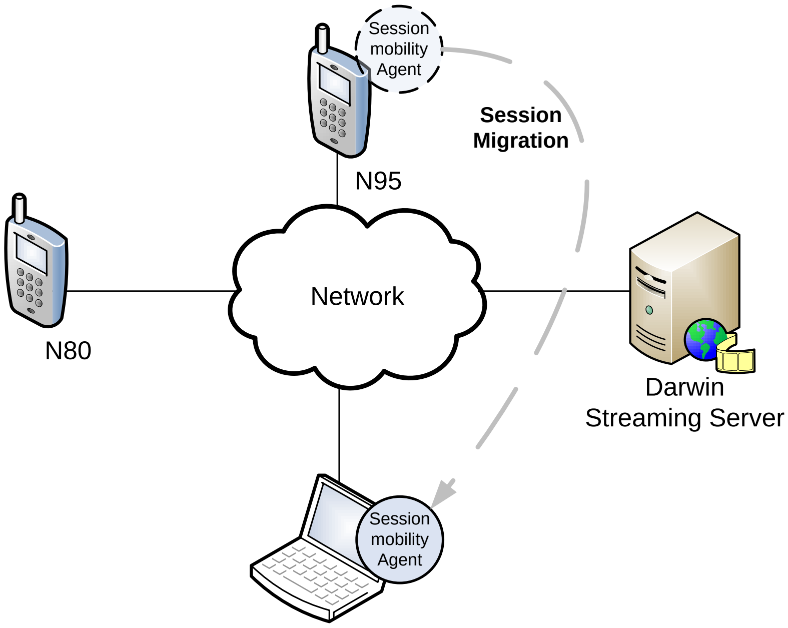


Figure 5.6: Test setup of the "multimedia session" use case.

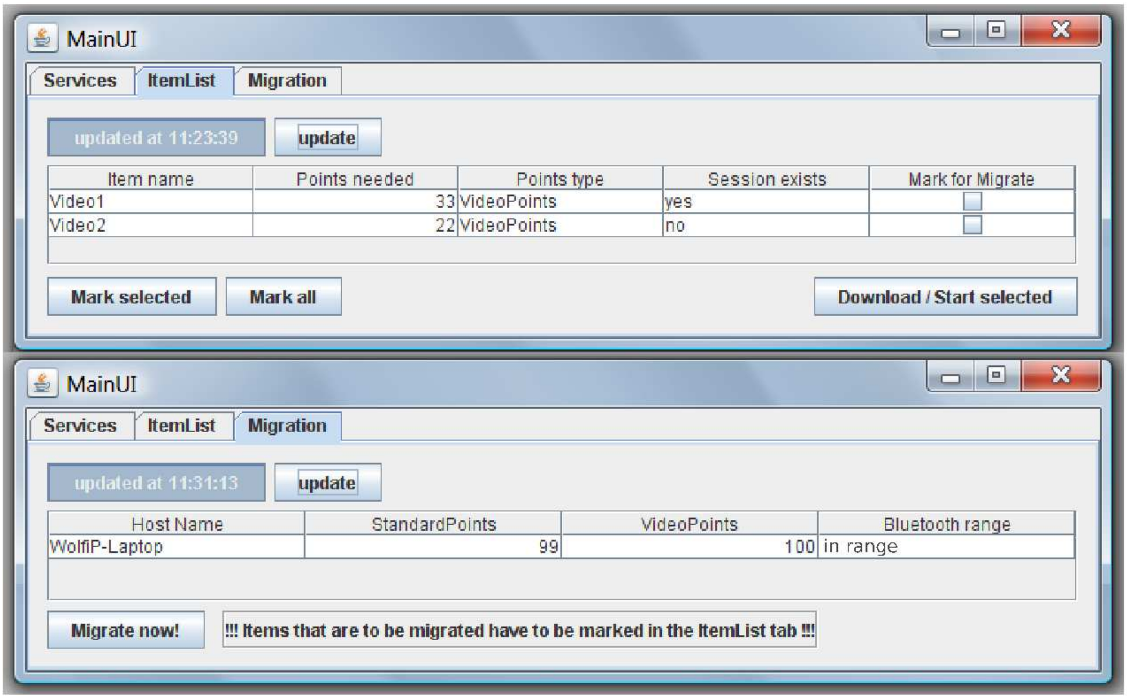
[1] Darwin Streaming Server – http://dss.macosforge.org/

information was saved as an MPEG-21 Digital Item. By evaluating the benchmark results of the available devices, the laptop was chosen as the migration target due to its superior benchmark results compared to those of the other smartphone (N80). After the session has migrated to the laptop, the higher quality version of the video was continued at the very spot it was paused on the cell phone. Figure 5.7 shows the manual migration mode where a DI (on the upper screenshot) and a migration target (on the lower screenshot) are available for selection. The DIs are listed with their name, recommended points for the playback and a flag for their session status. For target selection the device's name, overall score (standard points), session specific score (in this case video points) and reachability information is shown. While both versions of the video were referenced within the Content Digital Item, the capabilities of the laptop resulted in the selection of the higher quality version, since the smartphone was only capable for the playback of the low quality version.

5.1.6 Related Work

Several attempts have been made to satisfy the users needs for mobility and flexibility in their daily work. A recent and very simple form of session mobility can be obtained by using portable software on USB-sticks (e.g. The PortableApps Suite). All programs that should be ”mobile” available have to be installed directly on the USB-stick. However, while this approach is quite practical on a PC-platform, it can not be used with most mobile devices, and especially not with mobile phones. Moreover many applications are not suitable for installation on a USB-stick.

In [Min06] a streaming system for ubiquitous environments has been defined. Based on



Item name	Points needed	Points type	Session exists	Mark for Migrate
Video1	33	VideoPoints	yes	□
Video2	22	VideoPoints	no	□

Host Name	StandardPoints	VideoPoints	Bluetooth range
WolfiP-Laptop	99	100	in range

!!! Items that are to be migrated have to be marked in the ItemList tab !!!

Figure 5.7: Screenshots of the manual session migration interface.

the use of MPEG-21 it provides a mechanism for session mobility. As a result, users are able to continuously consume media through several terminals seamlessly. The system is able to transcode media or adapt it to the user's environment. Although this proposal seems to be very similar to AbSM, several major differences such as the absence of an automatic target host selection in [Min06] can be identified. Another very similar approach, the Ubiquitous Multimedia Framework for Context-Aware Session Mobility (UMOST), is presented in [Mac07]. It provides seamless user-level handoffs for multimedia sessions and quality of service management based on the use of MPEG-21. Adaptive streaming is also possible. In comparison with our proposal, these systems focus on streaming media and are only capable of transferring ”media” sessions. For parsing metadata, managing DIs and user sessions, a central server is needed.

The Internet Indirection Infrastructure (i3) is used in [Zhu05] to achieve Personal Mobility. It suggests using bluetooth-identifiers to locate devices. Every device that is within reach of a person's bluetooth-identifier is registered with its i3 trigger on a central registration server. If multiple devices are within reach, an internal protocol will decide which one will handle the current session. Compared to AbSM, there is no application-/ session-specific algorithm that will decide which device to use, and the user cannot choose a target device manually. Furthermore, the usage of applications not designed according to the client-server model is not possible, because of the lack of a direct session transfer between the originating and the target host.

Several Session Initiation Protocol (SIP) based systems for mobility of multimedia applications have been proposed. In [Sch00] the main scenarios are the re-routing of calls and the streaming of video, depending on the location of the user. [Ban06] proposes an architecture for mobility management, which supports soft handoff for voice and video streams. While these proposals certainly share some features with AbSM, they are focused on streaming media and there is no task-based algorithm that helps to decide which device to use.

Another SIP based system has been proposed by Adeyeye et al. [Ade10]. In this approach SIP is used to enable HTTP session mobility between two web browsers. Compared to our approach, a SIP application server is needed as a central component. Furthermore, so far the capabilities of this approach are limited to the ”browsing session” type.

In [Cye04] a distributed multimedia communication framework and conferencing software is presented, in which session information is managed within a LDAP directory server. This work uses a central server for session management and focuses on multimedia content and video streaming in an e-learning environment.

[Muk07] presents a way to enhance Universal Plug And Play (UPnP) in a home network by using session mobility. This approach allows for the transfer of multimedia sessions using the UPnP Audio Video Architecture (AV) between different UPnP enabled devices. Discovery and selection of a target device is done manually by the user from among the mechanism provided by the UPnP Framework. Although this approach seems to be very promising, it is still limited to UPnP AV and multimedia sessions.

A completely different approach is presented in [Mac07]. By using thin client technologies, namely the X Window System (X11) and Virtual Network Computing (VNC), a whole virtual desktop or just mobility enabled applications, can be moved from one device to another. A VNC server and X11 are mandatory components of this system. In contrast to our approach, a mobile application is located and executed on a server. As a result, communication costs and link failures may become severe issues.

The browser session preservation and migration (BSPM) infrastructure described in Song et al. [Son02] makes it possible for the user to switch devices while browsing the web and continue the same active web session on the new device. It uses a proxy server to store a snapshot of the browser session. In comparison to AbSM, BSPM is only applicable for browser sessions and does not take the capabilities of the target device into account.

5.1.7 Conclusion

In this section we presented a novel architecture for supporting session mobility. Through the use of the agent platform JADE, current user sessions may easily be carried by an agent from one device to another. Based on MPEG-21, a standardized way has been found to describe the session data and its context. Aside from the architecture, which is usable for different session types, a strategy for the selection of the "best" device for continuing a video session and a mechanism for migrating the actual session to the target device have been realized.

To expand AbSM, the support of other standards for device profiles like User Agent Profile (UAProf) [Ope06] and other profiles based on sources such as the free open source project WURFL[1] can be used. WURFL provides detailed profiles for more than 13.000 different mobile devices. Based upon a deeper knowledge of the device's technical capabilities, more enhanced strategies for the target device selection can be formulated.

5.2 TV Add-On Services (Intertainment)

The term interactive TV (iTV) is used for television systems in which the audience can interact with the television content. Interactivity in TV is often understood solely as the possibility of changing the storyline of a program. Besides this interpretation, iTV in general also means that TV add-on or TV related content and services are provided to the viewer (cf. section 2.1). For example, the chance to participate in a game show, gather additional information on news topics or directly buy a product presented in a commercial. In this context, we have developed a prototypical concept for iTV services. The deployment of new, innovative services is facilitated by the combination of digital TV and modern set-top boxes. For a detailed discussion of iTV, set-top boxes and other iTV related topics the reader is referred to chapter 2 section 2.1. In contrast to different iTV standards, our platform uses mobile devices to support multi-user and personalized access. The mobile devices are connected to the set-top box by an ad-hoc service mechanism. This connection is established using an existing home network environment. The use of an ad-hoc service architecture also enables inexperienced users to access and to use the services without having to worry about configuration. This guarantees easy access to these services for all users and accounts to the demands of user-centrism.

The rest of this section is organized as follows: Section 5.2.1 introduces the ad-hoc service mechanism used in our concept. In section 5.2.2, a description of our prototypical iTV service platform is given. Based on this platform, a service which allows for the

[1] WURFL – http://wurfl.sourceforge.net/

synchronization of additional content to an existing TV program, has been developed. Within this section, we provide details about this service and the components on both the server- and client side of our system. The platform has a large number of use cases. In section 5.2.3, we take a close look at two use cases that have been implemented, the ”news ticker scenario” and the ”game show scenario.” These use cases give a good impression of the capabilities of our platform. Section 5.2.4 gives a short overview of similar projects and products that offer mobile interactive services. Finally, section 5.2.5 concludes the discussion of our framework with a short summary of our work and its future implications.

5.2.1 UPnP

Universal Plug and Play (UPnP) [UPn06] is specified and maintained by the UPnP Forum[1], an industry initiative of about 900 members. Specifications for new devices and services are also certified and maintained by this forum. Thus, several specifications for different devices such as printers, scanner, home appliance and audio/video devices (UPnP AV) are available. UPnP is mainly intended for use in home networks. It makes use of different web technologies, such as HTTP, SOAP and XML, to offer a seamless and easy way to provide services and use them on different devices in local networks. Generally, UPnP is a kind of ”plug-and-play” concept for networked devices and services offering zero-configuration, transparent networking and automatic announcement and discovery.

The main building blocks of the framework are devices, services and control points:

- UPnP Device: A UPnP device can be understood as a container (root device) for nested logical devices and services. Devices can be implemented as a software component or directly as a physical device. For instance, a media center device may consist of a Blu-ray drive and several TV tuners, made available as services or even as embedded devices. In general, embedded devices do not depend on their root device and are therefore self-contained devices. Each device requires a unique device name (UDN), a device type, a name easily read by humans and a description. Device details, such as manufacturer information (vendor name, model, serial, etc.) and a list of its embedded devices and offered services, are described using an XML Schema and made available by the descriptive URL of the device. Additionally, each service requires its own unique service ID and a service type. Furthermore, icons can be made available for visualizing devices on a GUI. The device’s presentation URL points to a HTML page used for describing it, showing its status and providing a feasible way to enable an HTML-based control interface.

- UPnP Service: UPnP services are the main functional units in the system. Functionality is offered, based on XML, in the form of action definitions with their related arguments, state variables (at least one) and their associated data types. State variables may also be used as an event notification mechanism by specifying an ”eventing” type for sending event notifications on state changes. This can be done in two manners, via multicast or via single cast. Services provide a description URL, an event URL (eventSubURL), which is used in the UPnP eventing mechanism, and a control URL, used for action invocations by UPnP control points.

[1] UPnP Forum – http://www.upnp.org/

- UPnP Control Point: A UPnP control point is the main connector to the UPnP framework. It is capable of discovering and controlling services and devices in the network. After the discovery phase, it is used to retrieve device and service descriptions, invoke actions, subscribe to state-change-event notifications and process arriving event notifications.

Figure 5.8 shows a structural overview of the UPnP stack and the main phases of the UPnP collective work flow. These phases will be further discussed below:

- Addressing: Within this phase, network addresses are assigned to each device or control point. Generally, both versions 4 and 6 of IP are supported in UPnP. Different mechanisms for obtaining an IP address are used. First, the assignment is attempted via a DHCP service. If this fails, an automatic step for IP addressing, AutoIP, is commonly used. In Auto-IP, the first step is the implementation dependent, pseudo-random selection of an IP address, in the range of 169.254.1.0 to 169.254.254.255. After address selection, the Address Resolution Protocol (ARP) is used to verify if the address is currently in use or if it is free. If the address is already in use, the process starts again by selecting another address. In the case of automatically assigned addresses, a periodical check for an available DHCP service is conducted. Addressing is concluded with the successful assignment of an IP address.

- Discovery: After addressing, control points and devices are connected to the network. Through the use of the Simple Service Discovery Protocol (SSDP), a new device is able to announce its presence and advertise its services via a multicast message to all available control points. Aside from common information about the device, the unique identifier, the type and an URL to detailed information about the device is also released via the SSDP messages. This announcement is revoked if a device is going to be removed from the network. The same mechanism is used to discover

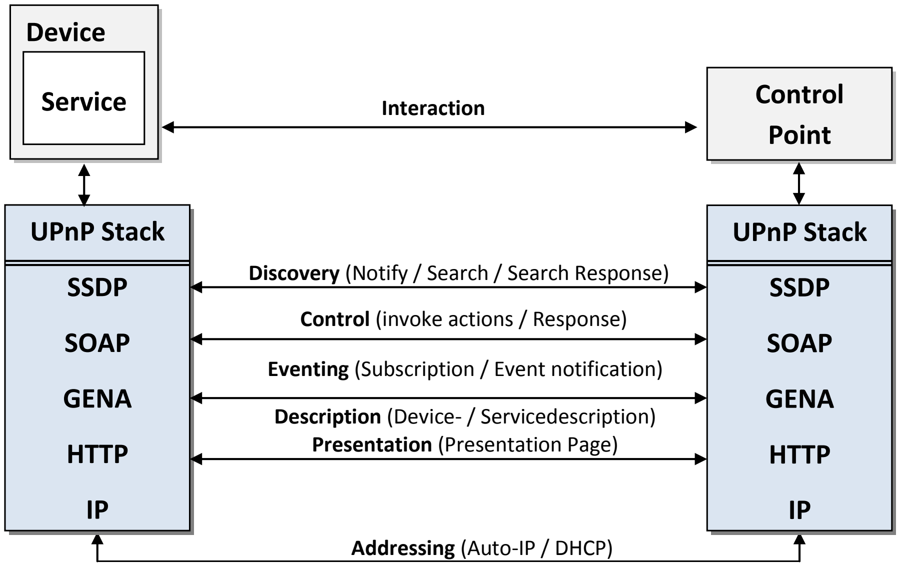

Device
Service
Interaction
Control Point
UPnP Stack
SSDP
SOAP
GENA
HTTP
IP
UPnP Stack
SSDP
SOAP
GENA
HTTP
IP
Discovery (Notify / Search / Search Response)
Control (invoke actions / Response)
Eventing (Subscription / Event notification)
Description (Device- / Servicedescription)
Presentation (Presentation Page)
Addressing (Auto-IP / DHCP)

Figure 5.8: The Universal Plug and Play (UPnP) architecture and protocols.

devices and services of interest, via multicast search messages using a control point (e.g. when its added to the network). For device and service discovery, search queries may be restricted to a specific IP address, a certain device, or service attributes such as service type or name.

- Description: After the discovery phase, a control point has only very limited information about the device and its services. In the description phase, a control point accesses the detailed device information via the device's description URL. This URL points to a detailed, XML-based description. This description contains information about the root device with all of its embedded devices, and details about all services offered and their capabilities. In most cases, the description is retrieved via a standard HTTP-Get request by the control point.
- Control: After retrieving and processing the XML description, all of the information needed for selecting and controlling a specific service is available at the control point. Aside from all available actions and their related input and output variables, also a list of state variables is included in the description. Based upon the concept of remote procedure calls, a control point sends action requests via HTTP posts to the fully qualified control URL of a service. After the completion of the action, the control point retrieves the action's results. Based upon the Simple Object Access Protocol (SOAP), the interaction between service and control point takes the form of SOAP XML messages. Thus, the interaction is platform, hardware, programming language and vendor independent.
- Eventing: Control and Eventing are closely related phases. In UPnP, control points may register for state changes of services. Subscription, as well as eventing is done through the use of the General Event Notification Architecture (GENA). Eventing is available in two main variants: The unicast event mechanism, where a control point subscribes to events for a specific state variable. The multicast event mechanism, by contrast, is used for announcing state changes to an IP multicast address. This mechanism is especially useful when multiple control points or controlled devices need to be informed about state changes. Event messages are expressed in XML and contain the names of the state variables and their current values.
- Presentation: Based on the presentation URL of a device, a feasible way to present and control devices can be offered. Depending on the specific capabilities of this page, information about the device varying from scant to complete or an interface for monitoring the device's status and controlling its services may be made available to the user. Aside from standard HTML and XHTML elements, different scripting languages or browser plugins may also be used for the device's presentation page.

Generally, security in UPnP depends solely on the protection of network access. Due to its open structure and concepts, it does not specify authentication or authorization mechanisms e.g. handling access rights, or for restricting the use of devices and services to a certain user or user groups. Nevertheless, based upon the Device Security and Service Console specification, a way to secure UPnP is available. Compared to other service oriented architectures, such as the Java Intelligent Network Infrastructure (JINI), Digital Living Network Alliance (DLNA) or web services, UPnP fits perfectly into our intertainment concept and the scenario described in this section. It focuses on the home entertainment and home network domain without restricting its application to a certain sub-domain in the way DLNA does. DLNA focuses exclusively on multimedia systems. Compared to JINI and web services, devices as well as services may be designed directly for the end-user.

5.2.2 Description of the System

Our prototypical system is mainly implemented using Java, which makes the system platform independent. It is divided into two parts: the service provider side and the service consumer side. This distinction stems from the principles of ad-hoc service architecture - the UPnP architecture (cf. section 5.2.1) - that forms an important part of the system. The service provider side is composed of UPnP devices and services, which are available in the home network environment. The service consumer side of the system - the user with his mobile device - uses a UPnP control point to discover the services and to take control of them. In short, UPnP can be described as an extension of the Plug and Play concept for networks. The main advantage of UPnP is that services offered can be used without further configuration. It hides the complexity and heterogeneity of the entire system and of the home environment. Thus, it fits perfectly with the philosophy of user-centrism. For a detailed description of UPnP, see section 5.2.1. Based on this architecture, a service for interactive TV and a control point, providing the intended interactivity, have been developed.

To the best of our knowledge, our iTV platform is the first platform that makes use of the UPnP architecture for interactive TV services. An important feature of the iTV platform is its support of many different types of mobile devices: smartphones, handhelds, UMPCs, and others. Compared to other systems that offer interactivity like MHP [Mor05], the interactivity in our system is not restricted to the TV set-top boxes, but is also available for the personal devices of the users. This enables a typical dual screen approach, where the display of the mobile device is used to display iTV applications in addition to the primary screen - the TV set. Furthermore, this allows our platform to support multi-user access.

Figure 5.9 presents an overview of the intertainment platform's architecture. The components in the figure are divided into two main groups: the service consumer components and the service provider components. In the following we will describe briefly each component and the associated flow of information.

Service Provider Components

- Video Disk Recorder[1] (VDR): We use this open source media center program to link the TV content to the other system components. The VDR enables a Linux PC to function as a digital receiver and video recorder. In general, any extendable media center software can be used.

[1] Video Disk Recorder - http://www.tvdr.de

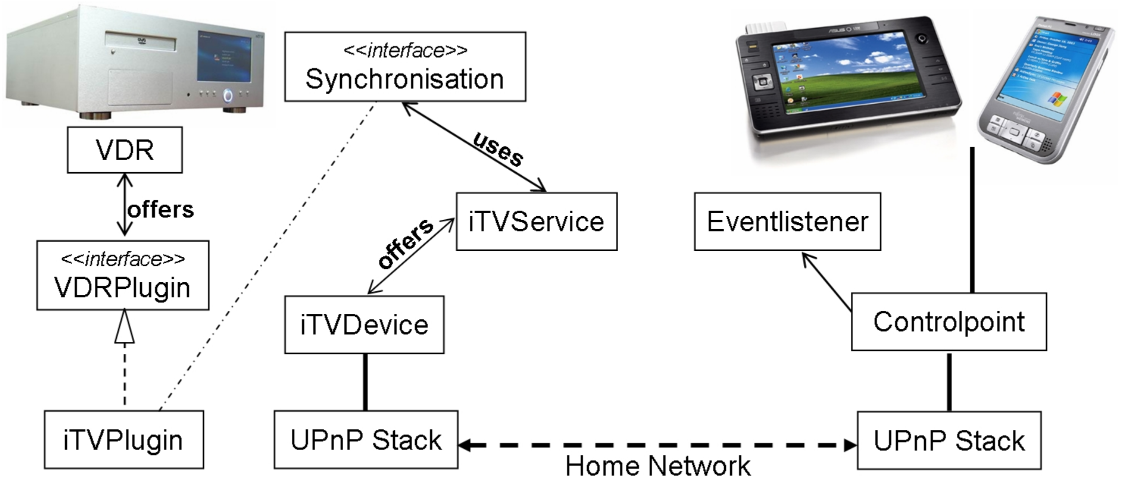


Figure 5.9: Overview of the intertainment demonstration platform architecture.

- iTVPlugin: This component is used as an implementation of the plugin interface of the VDR. Its main purpose is to extract synchronization information and build the synchronization timestamps. Based on these timestamps, TV content and related iTV applications can be synchronized. Two modes are supported in the plugin, the *livestream*- and the *playback*-mode. In the first mode, the timestamps are directly extracted from the MPEG-2 Transport Stream (cf. [Rei05]). These timestamps are delivered in the form of synchronization timestamps, the so-called presentation timestamps, in the packetized elementary stream. Based on the replay status of a recording, the second mode calculates the timestamps by evaluating the actual video frame and the associated frame rate. After the start of the media center, the iTV-plugin can be configured to extract synchronization information in the *livestream*-, in the *playback*- or in both modes. If the user watches live-TV, the presentation timestamps are combined with the name of the channel and the Video ID of the stream to allow for proper identification of the corresponding content. In *playback*-mode, the program name and the timestamps suffice for this identification function. The synchronization information is forwarded to the iTV device via a local socket connection.

- iTVDevice: This device is the first component of the *UPnP* framework in our system. It represents a UPnP root-device, and offers UPnP functions such as announcing the device and its services in the network, responding to search queries, sending event messages on state changes and controlling the device. The device forms a container for the service, as defined by the UPnP standard.

- iTVService: The iTVService implements the majority of the iTV functions. Within this service, the synchronization information provided by the iTVPlugin is processed and linked to the corresponding additional content. This content describes the presentation and interactivity related to specific events. A linking step is done by retrieving the corresponding content for the current synchronization information. For this purpose, we provide an XML-based event description that can be broadcast or provided separately. For the retrieval, this content is stored locally in a MySQL database (shown in figure 5.10). Each additional piece of content has an event type, a content type, a value, a start time and an end time. Content is often related to a scene. A scene is commonly defined by its start and end time, and therefore covers a specific part of the TV content. For now, only the content-types ”notification” and ”web-based application” are supported, but an expansion to other types could be easily integrated. The system makes use of several event types, which correspond to different purposes. The event types are divided into two main categories - ”scene-related” and ”scene-restricted” events. The content of a ”scene-restricted” event is valid only for the current scene, for example the current question of a game show. A ”scene-related” event corresponds to the current scene, but still remains valid afterwards. An example of this is the additional information on a news ticker topic.

- PresentationPageHandler (not pictured): A UPnP device can be presented and controlled through a presentation page. The iTV presentation page offers a web page with a short description of the iTVService.
- UPnP Stack: The Stack implements the functions of the UPnP framework.

Service Consumer Components

- Mobile Device: Various kinds of mobile devices like smartphones or handhelds are supported by the intertainment system. The only requirement is Java-support, either Java 2 Microedition (J2ME) CDC or Java 2 Standard Edition (J2SE).
- Control Point: The control point represents the main component on the client side. It implements a typical UPnP control point. Its purpose is to search for UPnP devices

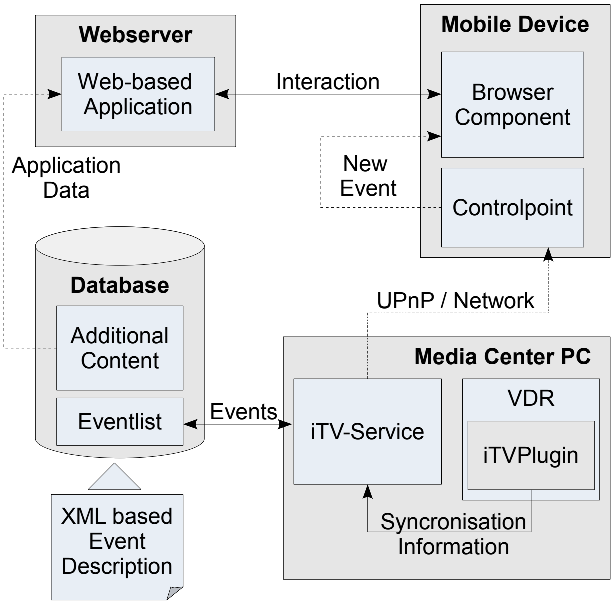


Figure 5.10: Flow of information in the intertainment demonstration platform.

and services in the network, and to make them available to the user. On startup, the discovery process of UPnP devices and services is initiated automatically. Every discovered device is shown on the user interface with its name and its associated icon. Further information on devices and their services can also be accessed. If a TV device with an iTV service is found, the control point automatically subscribes to iTV events.

- EventListener: This listener takes care of the synchronization of TV and additional content on the client side. At the time, which corresponds to the synchronization timestamps when new content becomes available, the EventListener is notified by event messages from the iTV service. Depending on the type of content and event, a reaction is released e.g. a notification is shown to the user or a web-based application is opened. After the event end time, the ”scene-restricted” content is not valid anymore and for that reason, removed and no longer accessible to the user. ”Scene-related” content may, but need not be accessed by the user after the end time. If two events overlap in time, the newer event will be brought forward.

Flow of Information

In figure 5.10, the flow of information is displayed. Initially the iTV control point on the mobile device detects the iTV-Service that is offered by the use of the UPnP framework. It establishes a connection to the iTV Service. The iTV Service receives two types of information: synchronization information from the VDR-Plugin and a list of events from an event-database that holds only events concerning the upcoming TV program. The iTV Service notifies the iTV control point whenever an event is triggered. Events with the content-type ”web-based application” cause the mobile device to open the related interactive web application. These applications are hosted on a web server, which runs locally on the set-top box or in the internet. With a local web server, there is no need for an internet connection. Nevertheless, an internet connection allows a back channel to the content provider which, for instance, can be used to provide most recent additional content or to collect usage statistics. Accordingly, the mobile device loads the application in its browser component. The data used by the application is also held by a database, which may be the same database that already holds the event data.

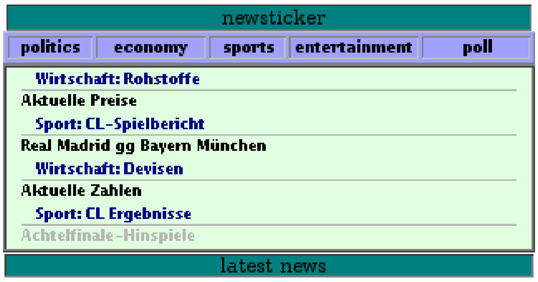
As explained above, multiple mobile devices can be used simultaneously. The interactive applications have to be designed in such a way that new devices can participate at each synchronization time stamp.

5.2.3 Use Cases

The next two paragraphs briefly describe two use cases of the intertainment platform. In both scenarios, web-based applications have been used to provide an attractive user interface to the iTV application.

The ”News Ticker Scenario”

News television networks often use news tickers to present headlines, weather information or stock prices. We have put into place a news ticker application that presents details



Wirtschaft: Rohstoffe
Aktuelle Preise
Sport: CL-Spielbericht
Real Madrid gg Bayern München
Wirtschaft: Devisen
Aktuelle Zahlen
Sport: CL Ergebnisse
Achtelfinale-Hinspiele

Figure 5.11: Screenshots of the ”news ticker scenario.”

related to the current information shown by the news ticker. The application also makes it possible to browse through all headlines. For the purpose of clarity, the news is organized into categories, like sports, stocks, etc. In figure 5.11 the news ticker application is shown as it would be displayed on a mobile device.

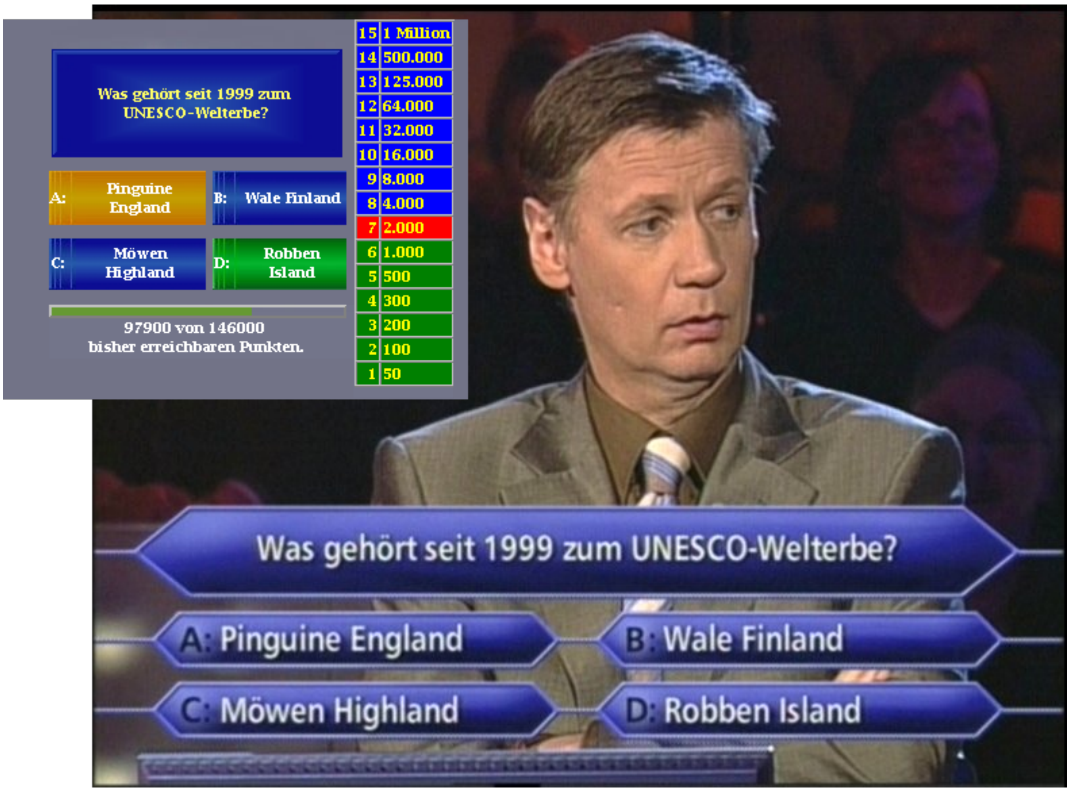
The ”Game Show Scenario”

Figure 5.12 shows our scenario based on the game show ”Wer wird Millionär?” (the German equivalent to the British show ”Who wants to be a Millionaire?”). The show consists of several rounds. In each of these rounds, a multiple choice question with four possible answers is asked. In our application, the viewer gets the same question, and can submit an answer using his mobile device. The answer can be modified as long as no solution is displayed on the TV-screen. The viewer receives feedback, simultaneous to the TV game show contestant, about his answers and statistics about his previous answers on his mobile device. The user is able to enter the game at any point of the show. On startup, the application always shows a short introduction and then starts with the current question. If more than one user is viewing the game on the same set-top box, each user plays in his own, independent game session. Since mobile devices have very different screen resolutions, the level of details and the amount of information displayed is adapted accordingly. The smaller screen shot in figure 5.12 shows the level of details on a UMPC with a screen resolution of 800 x 600 pixels. The version of this screen adapted to the smaller resolution of a PDA does not display the statistics. On a smartphone, the score is omitted as well.

5.2.4 Related Work

Several projects and standards deal with interactive TV in various forms, and to varying levels of success. The Multimedia Home Platform specified by the MHP group, a subgroup of Digital Video Broadcasting Project[1] (DVB), is one of the most prominent platforms. MHP provides a hardware and vendor independent execution environment for digital applications and services in the context of DVB standards. MHP applications range from very simple applications with rudimentary interaction capabilities, to powerful and complex applications. Typical applications are news tickers, stock tickers, advanced

[1] Digital Video Broadcasting Project – http://www.dvb.org/



Was gehört seit 1999 zum UNESCO-Welterbe?

A: Pinguine England

B: Wale Finland

C: Möwen Highland

D: Robben Island

97900 von 146000 bisher erreichbaren Punkten.

15	1 Million
14	500.000
13	125.000
12	64.000
11	32.000
10	16.000
9	8.000
8	4.000
7	2.000
6	1.000
5	500
4	300
3	200
2	100
1	50

Was gehört seit 1999 zum UNESCO-Welterbe?

A: Pinguine England

B: Wale Finland

C: Möwen Highland

D: Robben Island

Figure 5.12: Screenshots of the "game show scenario" – Question: "What is a UNESCO world heritage site since 1999?"

teletext, educational and E-commerce services. Other standards like the OpenCable Application Platform (OCAP)[1] or the Advanced Common Application Platform[2] (ACAP), which are widely used by US TV system operators, offer capabilities comparable to those of MHP. For a comprehensive overview of theses approaches, the reader can refer to section 2.1.3. A common characteristic of these platforms (also of recent developments such as HbbTV) is the clear focus on TV set-top boxes. Interactive applications are exclusively run using the set-top box. The user may interact with the system by using a remote control or a special keyboard. For these applications, the screen of the TV-set has to be shared between the user interface and the current TV program. The support of multiple users in such an arrangement is not possible. Other research projects and standards address the area of mobile TV services, in which the mobile device is used as a replacement for the TV-set, like the Savant project [Rau05] or the DVB-H Electronic Service Guide (ESG) [Eur06b]. The Television and Mobile phone Assisted Language Learning Environment (TAMALLE) [Fal05] facilitates language learning by combining an iTV learning application with a mobile phone. Using a mobile phone, the learner can access the summary of a program as well as a list of different language items that may appear inside a program.

Commercial projects have also discovered the great potential of TV add-on content and

[1] OpenCable Application Platform – http://www.opencable.com/ocap/
[2] Advanced Common Application Platform – http://www.acap.tv/

services on mobile devices. The business model of these projects focuses on advertisement and chargeable services. The Service2Media Company[1] offers several services like an electronic program guide or interactive advertising on mobile devices. BLUCOM[2], a former discontinued product of the Astra Platform Service Company, made use of mobile phones for interactive TV related applications presented in a special browser. The content was offered to the mobile device by a special set-top box, via bluetooth or was downloaded via a cellular network.

Another system called Betty[3], offered by Betty TV, used a special remote control with a small greyscale LCD-display. Additional content is displayed directly on the screen of the remote control. The Betty system only offered very simple services.

In contrast to these products, our proposal is based on open source software and, thanks to the use of UPnP, features easier extendability and higher flexibility.

5.2.5 Conclusion

The intertainment platform presents an easy way to provide synchronous TV add-on services and interactivity on personal mobile devices. The UPnP standard allows an easy extension of our system with other services like remote control functionality and recommendation systems. As proof of these concepts, two use cases - the ”game show scenario” and the ”news ticker scenario” - presenting the capabilities of our platform have been demonstrated on the ACM MM 07 in Augsburg [Höl07]. Furthermore, our system has been successfully presented on the CeBit 2007 in Hannover and has been published in the IEEE Multimedia Magazine [Höl08].

[1] Service2Media - Mobile Interactive TV – http://www.service2media.com/

[2] Blucom – http://www.blucom.de/

[3] Betty TV AG – http://www.betty-tv.de/

CHAPTER 6

Personalization of Multimedia Consumption

When defining user-centric multimedia, it is important to emphasize the personalization of the consumption and the usage of content and services that are essential parts of such a multimedia system. Especially in the realm of television, the demand to cater such systems to the needs of the audience is constantly growing. As digital television becomes more widely used and the number of channels increases, the audience is confronted with a myriad of program choices. At the same time, the amount of program information offered in electronic program guides (EPGs) such as Guide Plus+ from Gemstar[1] or TV Movie[2] from Heinrich Bauer Verlag is growing as well. Overwhelmed by this large variety of programs and related information, many viewers give up selecting programs systematically. As a result they get around by zapping, asking for others' recommendations, or always watching the same programs or channels. However, several systems have been developed with the aim of helping users to cope with this issue by assisting them in their program selection. Most of these systems focus on guiding the consumer through the considerable amount of data. They provide search and filter functionalities and display a clear program overview. Enhanced systems (e.g. the personal video recorder TiVo[3] in the United States) employ collaborative filtering techniques to generate recommendations based upon the interest of users. This is accomplished by rating the degree of similarity of a program to "likes" and "dislikes" identified by the user. In the near future, several new competitors with interesting approaches in the area of intelligent TV guides will enter the market (e.g. in Germany MY Personal TV Digital[4], Guide Plus+ as well as TiVo).

In contrast to traditional TV Guides and their ways of organizing program overviews, online services and platforms frequently make use of tags to enhance searchability and navigation. Tags are commonly known as descriptive and personal keywords for any item in its concrete context. Platforms such as Delicious[5], which enable the user to add tags to URLs he shares, and Last.fm[6], which uses tags for the purpose of navigation and of

[1] TV Guide Plus+ - http://www.europe.guideplus.com/
[2] TV Movie - http://www.tvmovie.de/
[3] TiVo - http://www.tivo.com/
[4] MY Personal TV Digital - http://mypersonaltvdigital.net/
[5] Delicious - http://delicious.com/
[6] Last.fm - http://www.lastfm.com/

generating recommendations, are just two prominent examples of the many successful collaborative tagging systems. Collaborative tag collections are also often referred to as folksonomies.

In this chapter, we present an approach which addresses the audience's problems in the realm of TV watching by providing them with individual program recommendations and well-arranged program overviews on their personal remote control. This approach combines content-filtering methods, inspired by the success of various methods of fighting spam, state-of-the-art classification techniques and mechanisms well known from the area of information retrieval and text mining, with concepts from the area of collaborative tagging systems for the recommendation generation. Recommendations are mainly generated based upon the analysis of an individual user's history of watched and tagged items and, optionally, upon his or her explicit profile. The user's interaction with the system is monitored to create and automatically update an implicit profile. The combination of user actions and program metadata provided by an EPG are analyzed to identify latent semantic relationships between preferences and programs. Additionally, an explicit MPEG-7 compatible user profile has been integrated in order to overcome the cold start phase, which is one of the shortcomings of an approach that relies solely on the analysis of the user's behavior. Moreover, a collaborative filtering approach based upon tags is used for recommendation generation and to prevent the system from degenerating (cf. paragraph ”Challenges” in chapter 4).

The remainder of the chapter is organized as follows. In section 6.1, fundamental concepts and methods are introduced. A system overview and a detailed discussion of its main components are presented in section 6.2. Following this, section 6.3 and 6.4 describe the major conceptual parts of the system. Section 6.3 details our content-based media recommender. Our collaborative recommendation component is presented in section 6.4. Finally, section 6.5 concludes this chapter with a comprehensive experimental evaluation.

6.1 Fundamental Concepts and Methods

In this section, we introduce the fundamentals of classification based upon different content filtering approaches, as used in the context of spam, Support Vector Machines and Latent Semantic Indexing. Based upon the tokens or token groups supplied by the tokenizer, the classifier identifies implicit relations between multiple textual descriptions. Our tokenization approach will be introduced in section 6.1.1. A large number of previous observations, e.g. the history of programs watched or recorded by a user, is used to generate a prediction of desirable upcoming programs. Several different methods and concepts can be used to complete this task. Section 6.1.2 describes three statistical filtering approaches used to fight spam. A state-of-the-art classification mechanism often utilized in the area of pattern recognition is described in section 6.1.3. Section 6.1.4 introduces Latent Semantic Indexing, an approach commonly used for information retrieval tasks.

6.1.1 Enhanced Tokenization

In many systems, natural language inputs play an essential role. In our system, text descriptions are also of great importance, as multimedia metadata is at the foundation of our recommender system approach. Thus, the following questions regarding the preparation and processing of textual input need to be answered:

- What is the best way to process and segment such an input, for further processing steps?
- How can the input segments be further annotated and enriched with metadata concerning grammar, semantics and content?
- Which parts of the input are particularly important in relation to user interests?
- How could the system benefit from the usage of these parts and their extended annotations?

In the following section, we will present a framework for processing and analyzing textual inputs that facilitates the identification and extraction of important semantic concepts. For accomplishing this task, several techniques from the fields of Text Mining and Natural Language Processing (NLP) have been combined within this framework. For more information about NLP mechanisms used in this system component the reader may refer to chapter 3. Operating on program descriptions, the system has been optimized in a way specific to the TV domain. However, the framework can also be used with other text categories. The rest of this section is organized as follows. In the first part of this section, we will present a system overview and describe the individual components. In addition to this, different tools and libraries used in our Tokenizer framework and an example output will be briefly discussed. Finally, we will conclude this section with a short experimental evaluation of the enhanced tokenization system.

Overall Process

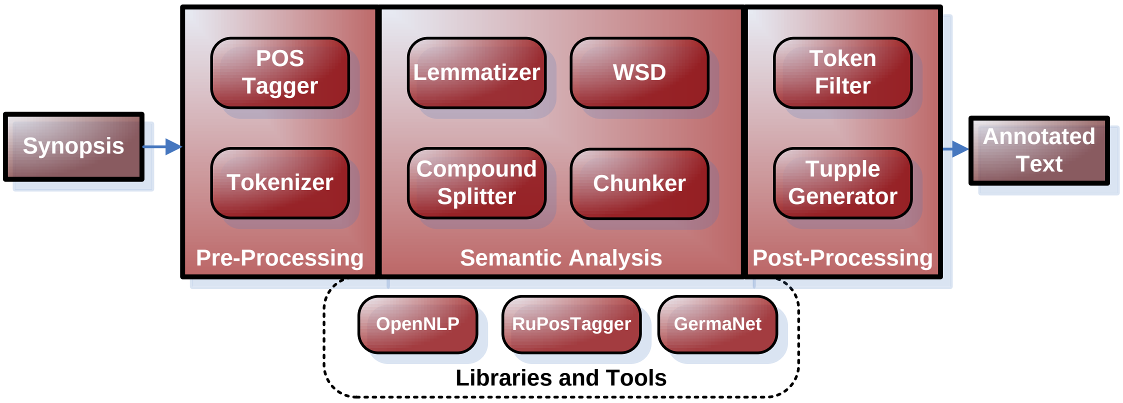


Figure 6.1: Architecture of the enhanced text tokenizer framework.

As shown in figure 6.1, the Enhanced Text Tokenizer Framework operates on textual inputs, in our case, the synopsis of a program descriptions in German. As illustrated in the figure, the entire tokenization and enrichment process is partitioned into the three following phases:

1. Preprocessing: The textual input first needs to be prepared for further analysis. Thus, in the preprocessing step the input is converted to an internal representation. Depending on the components to be used later in the semantic analysis step, special escaping and replacement of characters have to be done (e.g. several libraries have problems with URLs and need a special escaping for them).

   Subsequently, the input is split into its constituent segments through **tokenization**. Often, this step is integrated with the **POS tagging** step or at least strongly linked. In general, the system does not require the use of a specific POS tagger as long as the tagger uses the Stuttgart-Tübingen-Tagset (STTS) [Sch99]. Once the tokenization is done, the tokenized text with its POS annotations is forwarded to the semantic analysis phase.

2. Semantic Analysis: During this phase, the input is augmented with additional information. First, each token is processed by the **lemmatizer** and annotated with its lemma. If a certain word is ambiguous, it is possible to assign multiple lemma. In such cases, the POS tag is used to find the lemme which is most likely correct.

   In our approach **compound splitting** is done in a very limited way. This is due to the fact that many words can be decompounded in multiple variants where the correct decomposition is often not clear (cf. section 3.2.4). Thus, in this step, we only consider capitalized nouns. To avoid decomposition errors, the synsets of all constituent parts of the compound and their specificity (position in the hierarchy) taken from GermaNet are additionally used. Synsets commonly refer to a group of words that are considered to have the same meaning and refer to the same semantic concept. The decomposition is done only if the specificity of at least one part of the compound is higher than the specificity of the compound itself.

   After this, an ontology is used for semantically augmentation. In our approach, we make use of the wordnet derivate - GermaNet. It covers the most important word classes such as nouns, verbs and adjectives. For all tokens belonging to these classes, GermaNet is used to retrieve available synsets. These synsets are then added as annotations to the token.

   The meaning of most synsets for a lemma can be considered to be almost identical (in regards to specificity) but this does not hold of ambiguous words. Synsets located in regions far away from each other in the ontology are strong indicators for ambiguity. Because of this, the next processing step for such tokens is **word sense disambiguation**. Although several approaches for this task have been introduced in section 3.4.1, they are not applicable in our approach. Commonly these approaches need to determine the context of a token based on a gloss, a domain annotated corpora or at least on the frequency of different word senses in the actual domain. As none of the mentioned resources are available, we propose an algorithm reliant solely on lexical relations. In our algorithm, the contextual surroundings of the ambiguous token is taken into account up to the sentence boundaries. In an iterative process, closely related synsets of the ambiguous token are merged. Closely related synsets are in this case synsets in a father-son or sibling relation in the subsumption hierarchy. Then, the similarity between the token's synsets and the synsets of its unambiguous token context (tokens where $\#synset <= 1$ holds) is measured. All synsets of the ambiguous token with a similarity less than the token's overall average similarity are discarded. This process is repeated until the token's synsets are stable. Roughly explained, the similarities between the different senses of the ambiguous token and the surrounding tokens (with unambiguous senses) are used to identify which meaning is most closely related to the sense of the word, based on its context. Similarity is measured by the Leacock-Chodrow measure as described in section 3.4.2.

The **Chunking** step concludes the text analysis phase. The main goal of this step is to find consecutive tokens that are closely related to each other (such as ”Dr.” ”House”) and reorganize them into chunks. In our system, chunks are represented as composite tokens or as additional annotation which can be handled as atomic units. Especially in cases where tokens with different semantics are reorganized, it is not desirable to loose the constituent tokens and their annotations. Thus, a head token is identified which determines the main semantic of the chunk. The grammatical category and the associated synsets of the chunk are also defined by the head token. To facilitate the formation of chunks the following basic rules are used in our system:

- Phrases (PHR): Phrases are often included in GermaNet and therefore easy to identify using their own synset. Thus, token sequences such as ”Adam's apple” or ”A pretty penny” can be merged.
- Personified Profession (PPR): The PPR pattern is used to identify words typically composed by a profession and a name, such as ”Inspector Clouseau.” The profession, used as the head of the composite, can be easily identified based on the hyponym graph of the abstract concept of all professions.
- Named Entities (NE): Consecutive tokens tagged as named entities by the POS tagger such as ”Harry Potter” are merged. No token head is defined here.
- Attributive Adjective (ADJA) - Noun (NN) or Adverbial Adjective (ADJD) - Verb (V): In both, the ADJA - NN (e.g. ”beautiful girl”) and the ADJD - V (e.g. ”running fast”) case the tokens commonly can be merged. Consequently, the noun and the verb become the head of the chunk.

Due to the overlapping of rules (e.g. PPR and NE), it is clear that the execution order (similarly to the itemization order) has to be respected.

3. Post-processing and Tuple generation: The last phase of our tokenization process addresses the formation of the output and the extraction of semantically rich tokens (in our case, token pairs). For instance, the main essence of a plot can often be formulated into tuples made up of only a few words, such as ”car - crashs,” ”policeman comes,” and ”saves life.” Because the length of the token sequence commonly has a considerable impact on the performance of tuple generation, we decided to extract concepts based on token tuple. The selection of the appropriate tokens and their reorganization into tuples is done in the tuple generation step. Especially in German, the selection of semantic rich tokens is a challenging task due to the highly flexible

word order of the German language. Therefore, we propose the use of heuristic rules for this task. Our main assumption is that certain POS combination such as verb - noun (like in the previous example) or adjective - noun are more likely to carry semantically important information. Thus, in the first step, the tokens in each sentence are paired with all other tokens within a certain search window, as defined by the distance between words. In general it is assumed that words separated by a smaller distance are more likely to be related to each other than those with a larger distance. Among others, two different functions for measuring this distance were identified as applicable in our system:

- ”word count” distance: To measure the distance between Tokens $t1$ and $t2$ the distance score is increased by one for each token on the way from $t1$ to $t2$.
- ”POS-weighted” distance: Based on the idea that different grammatical classes (POS-tags) have different impacts on the distance between important tokens (e.g. an article is not as important as a noun), different distance weights can be assigned to each POS. For the distance calculation, each token’s weight is summed up on the way from $t1$ to $t2$.

Figure 6.2 illustrates an example for both distance measures with a search window of size 3. The POS-weight settings have been freely chosen for this example. As a starting point, the token ”Spaß” (eng. ”fun”) is used. Due to the different weights determined by a POS tag, the weighted method is able to find the verb ”‘hatten” (eng. ”had”) for a correct pair formation (eng. ”had fun”), whereas in the unweighted method the verb is not part of the context of the word ”Spaß.” After the pairing of tokens, a list of token pairs is available. In the pairing process, only tokens inside of a certain search window are considered. To reduce the number of possible token combination, a **filtering** is conducted before they are ranked based on our expressiveness heuristic. Pairs are filtered based on the following conditions:

- Some token combinations do not possess any specific meaning at all, such as combinations with punctuation marks.
- Tokens without any synsets are skipped. As GermaNet contains only the grammatical classes nouns, verbs, adverbs and adjectives, the number of pairs is strongly reduced.

Based on their rich annotations (synsets, POS, etc.), the expressiveness of token pairs is measured according to several parameters. For one parameter, the POS tag of each token is taken into account. This is done according to a POS-combination

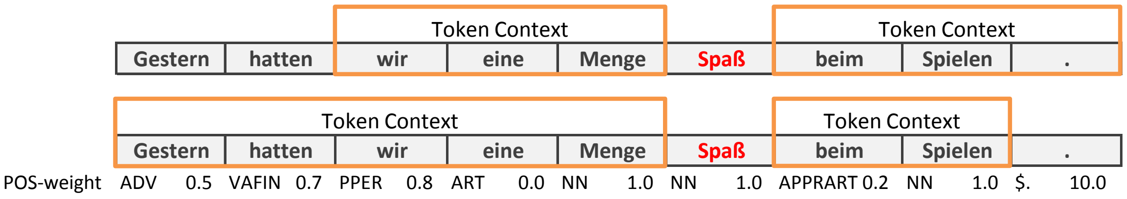


Figure 6.2: ”word count” vs. ”POS-weighted” word distance with an exemplary POS-weight setting.

weighting matrix. This matrix has been build with respect to the ratio between the distribution of POS pairs in the test samples and in POS-pairs considered to be relevant by the user (cf. table 6.1). The ratios used have been gathered in a preliminary evaluation step described in the paragraph entitled ”Experimental Results and Evaluation.” The distance between the tokens of the pair is also used for estimating the correlation between them. Therefore, the weight of the pair (score) is devaluated by dividing it by $1 + dist$. Furthermore, the number of synsets per token can be used as an indicator for the ambiguity of meanings. For instance, consider the word ”bank,” which can be understood as a financial institute or a river bank, and therefore features many synsets. Additionally, attributes like the token’s position in the sentence or its broader position in the text can also be used. Based on this, a score value in the interval [0,1] is assigned to each token pair. All pairs with a score exceeding a threshold signified as t are considered to be relevant.

As shown in figure 6.1, our framework makes use of several tools and libraries:

- POS Tagger and Tokenizer: Among others, the Open Natural Language Processing Project[1] (openNLP) is one of the most prominent resources for open source NLP tools and related projects. It offers tools for tasks such as tokenization, POS-tagging, Named Entity Recognition (NER) and Chunking. Most of these tools are available for English, German, Spanish and Thai. Whereas NER and Chunking are only provided in English. Thus, openNLP tools are used in our framework for tokenization and POS tagging. Examples of other similar tools that can be used for these tasks are the StandfordNLP and the TreeTagger.

- Lemmatizer: For lemmatization in our framework, we used the LemServer which is part of the RussianPOSTagger[2] (RuPosTagger). The lemmatizer is available for German, Russian and English. It is written in C++ and offers a Java wrapper and a simple XML-RPC API for remote access. Thus, it is easy to access and very fast. Alternatively, Morphy or LemmaGen can be used for this task as well.

- Ontology: GermaNet[3] is used as a word taxonomy in our system. It is hosted by the university of Tübingen and freely available for academic use. For work in the German language, it is frequently used and the most complete word net derivate.

Consider the program synopsis about the world’s largest model railway shown in listing 6.1 as an input example. An excerpt of the tokenization framework’s output is shown in listing 6.2. This output is structured as follows: First, all identified locations such as the city ”Hamburg” are listed. As is common, all tokens are mentioned with their related raw values, lemma and synsets. Furthermore, for composite tokens the partitioning into a composite head, if any, composite qualifiers and its constituting tokens is additionally added. Locations are commonly identified based upon their relation to the root synset of all locations. Afterwards, the named entities are listed. Subsequently, all extracted

[1] OpenNLP – http://opennlp.sourceforge.net/

[2] RussianPOSTagger – http://rupostagger.sourceforge.net/

[3] GermaNet – http://www.sfs.uni-tuebingen.de/GermaNet/

token pairs of interest are listed within the ”pairs” element in the XML file. Finally, the whole input text is listed and the analyzed text is organized based upon its constituting sentences.

Listing 6.1: Synopsis of the Program

```
Mitten in der Hamburger Speicherstadt sorgen 50.000 Bäume, 30.000 Figuren , 6500
Meter Gleis , 3000 Häuser und Brücken sowie jeweils 1000 Signale und Weichen für
eine Modelleisenbahn der Superlative , die nur durch modernste digitale Steuerungs-
technik beherrschbar bleibt . 60.000 Lämpchen beleuchten Häuser , Laternen und
Straßenzüge. Aus Holz , Gips und Kunststoff ließen die Erbauer ein hochalpines
Skigebiet , ein Mittelgebirge , Hamburg samt Hafen , Hauptbahnhof und AOL Arena ,
eine Bergbaulandschaft und unzählige liebvolle Details entstehen .
Jede Viertelstunde durchläuft die Anlage einen virtuellen Tagesablauf . Es dämmert ,
wird Nacht , dann erweckt die aufgehende Sonne die maßstabsgetreue Miniwelt erneut
zum Leben . Seit Juli 2000 basteln über 50 Männer und Frauen an der Erfüllung eines
3 Millionen Euro teuren Kindheitstraums : Den hatten die geistigen Väter und Erbauer
der Anlage , Freddy und Gerrit Braun , als sie im Urlaub in der Schweiz einen kleinen
Modellbahnshop in Zürich betraten . Seiher ließ die beiden Brüder die Idee nicht
mehr los , die größte Modelleisenbahn der Welt zu bauen .
```

Listing 6.2: XML Output of the Tokenizer

```
<AnalysisResult ID="Männertraum im Miniland">
<Locations>
 <Location value="Mittelgebirge" lemma="Mittelgebirge" synsets="nOrt.1552"/>
 <Location value="Hamburg" lemma="Hamburg" synsets="nOrt.2079"/>
 <Location value="Zürich" lemma="Zürich" synsets="nOrt.2079"/>
 <Location value="Landschaft" lemma="Landschaft" synsets="nOrt.1325"/>
 <Location value="Schweiz" lemma="Schweiz" synsets="nOrt.2444"/>
</Locations>
<NamedEntities>
 <NamedEntity name="Braun"/>
 <NamedEntity name="Freddy"/>
 <NamedEntity name="AOL Arena"/>
</NamedEntities>
<Pairs>
 <Pair>
  <AnnotatedToken value="Stadt" pos="NN" lemma="Stadt" synsets="nOrt.1886"/>
  <AnnotatedToken value="hamburger" pos="ADJA" lemma="hamburger"
      synsets="nOrt.2079"/>
 </Pair>
 <Pair>
  <AnnotatedToken value="Technik" pos="NN" lemma="Technik"
      synsets="nKognition.1173"/>
  <AnnotatedToken value="modernste" pos="ADJA" lemma="modern"
      synsets="aZeit.198,vVeraenderung.873"/>
 </Pair>
  <Pair>
  <AnnotatedToken value="Modelleisenbahn" pos="NN" lemma="Modelleisenbahn"
      synsets="nArtefakt.5193"/>
  <AnnotatedToken value="größte" pos="ADJA" lemma="groß"
      synsets="aAllgemein.3,aMenge.134"/>
 </Pair>
 <Pair>
  <AnnotatedToken value="Details" pos="NN" lemma="Detail"
      synsets="nKommunikation.92"/>
  <AnnotatedToken value="liebvolle" pos="ADJA" lemma="liebvoll" synsets=""/>
 </Pair>
 <Pair>
  <AnnotatedToken value="Miniwelt" pos="NN" lemma="Miniwelt" synsets=""/>
  <AnnotatedToken value="maßstabsgetreue" pos="ADJA" lemma="maßstabsgetreue"
      synsets=""/>
 </Pair>
```

```
</Pairs>
<AnalyzedText>
...
 <AnalyzedSentence size="25">
  <AnnotatedToken value="aus" pos="APPR" lemma="aus" synsets="aZeit.225"/>
  <AnnotatedToken value="Holz" pos="NN" lemma="Holz" synsets="nnatGegenstand.57"/>
  <AnnotatedToken value="," pos="$," lemma="," synsets=""/>
  <AnnotatedToken value="Gips" pos="NN" lemma="Gips"
      synsets="nnatGegenstand.253,nSubstanz.866"/>
  <AnnotatedToken value="und" pos="KON" lemma="und" synsets=""/>
  <AnnotatedToken value="Kunststoff" pos="NN" lemma="Kunststoff"
      synsets="nSubstanz.126"/>
  <AnnotatedToken value="ließen" pos="VVFIN" lemma="lassen"
      synsets="vAllgemein.349,vBesitz.24,vKognition.373,vVeraenderung.88"/>
  <AnnotatedToken value="Erbauer" pos="NN" lemma="Erbauer" synsets=""/>
  <CompositeToken value="hochalpines Skigebiet" pos="NN" lemma="hochalpin
      Skigebiet" synsets="nOrt.148,nOrt.1" string="hochalpin.Ski Gebiet"
      head="yes" qualifiers="1">
   <CompositeHead>
    <CompoundNounToken value="Skigebiet" pos="NN" lemma="Skigebiet"
        synsets="nOrt.148,nOrt.1" string="Ski Gebiet" head="yes" qualifiers="1">
     <CompositeHead>
      <AnnotatedToken value="Gebiet" pos="NN" lemma="Gebiet"
          synsets="nOrt.148,nOrt.1"/>
     </CompositeHead>
     <CompositeQualifiers>
      <AnnotatedToken value="Ski" pos="NN" lemma="Ski"
          synsets="nArtefakt.2312,nGeschehen.4640"/>
     </CompositeQualifiers>
    </CompoundNounToken>
   </CompositeHead>
   <CompositeQualifiers>
    <AnnotatedToken value="hochalpines" pos="ADJA" lemma="hochalpin" synsets=""/>
   </CompositeQualifiers>
  </CompositeToken>
 ...
</AnalysisResult>
```

Experimental Results

The evaluation of our tokenization framework has been done on a test corpus, where 150 program descriptions have been randomly selected. Taking this test data, our system has been used to process each text (as described at the beginning of this section) and to identify a list of possible interesting token pairs. In the pairing step, each token was combined with all tokens within a search window of 4 tokens, resulting in about 60 000 token pairs. To reduce the number of token pairs to a manageable size, a filtering step, as described in the previous paragraph, has been conducted. Due to the restrictions imposed by the use of GermaNet and the appearance of many irrelevant POS combinations such as punctuation marks with other tokens, the number of candidate token pairs was reduced to 4 300. For establishing a base line about the semantical importance of different token combinations, a test with several users has been conducted. Users have been asked to select the most significant token pairs out of all available pairs for each text and rank them accordingly. On average, 29 token pairs per text have been generated and were available for selection. Within an evaluation period of 3 weeks a total number of 73 users participated and 1286 text evaluations were made. On average, each text has been evaluated 8,6 times. Table 6.1 shows the adjusted occurrence of the percentages of different POS tag combinations, as selected by the users from among the semantically important tokens. Tag combinations with an occurrence count lower than 5 have been omitted during the evaluation phase.

POS token 1	POS token 2	Percentage
ADJA	NN	29.4 %
NN	VVFIN	20.7 %
ADJD	NN	7.5 %
ADJA	VVFIN	6.3 %
NN	VVPP	6.0 %
NN	VVINF	5.6 %
ADJD	VAFIN	2.8 %
ADV	NN	2.8 %
ADJD	VVFIN	2.5 %
VMFIN	VVINF	1.6 %
ADJA	VVPP	1.5 %
ADJA	VVINF	1.4 %
ADV	VVFIN	1.2 %
ADJD	VVPP	1.2 %
ADJD	VVINF	1.2 %
ADJD	ADJD	1.1 %
ADJA	ADV	1.1 %
other		6.5 %

Table 6.1: Quantities of different POS tag combinations in the evaluation results.

In examining this data, several interesting facts have been revealed. More than 50 % of the token pairs considered as descriptive were covered by only two POS combinations. Furthermore, almost all relevant pairs were within a word distance of 3, and nearly 50 % of them are neighbors. Thus, a search window of 3 can be considered sufficient for gathering almost all relevant token pairs.

To judge the overall performance of our system, and especially the ability to find descriptive pairs, the dataset gathered in the user study has been used as the base line. Several tests have been conducted, for different pairing thresholds ranging for 1.0 to 0.0. It was, however, at the threshold of 0.85 that the system showed the best performance. At this threshold, almost 80 % of the descriptive pairs identified by our user were found by the system.

6.1.2 Spam Classifier

In the context of fighting spam, different approaches ranging from hand written rule sets to black-, white- and greylisting mechanisms to content-based filtering methods have been applied in an effort to properly detect spam. The application of classification mechanisms by statistical filters is well known, and commonly used in this field. Among other methods, statistical (often Bayesian) filters are used in most spam detection system today. For a detailed discussion of different spam fighting approaches, interested readers may refer to [Zdz05]. The roots of this technique go back to 2002 when Paul Graham grew tired of constantly having to write new rules for separating spam from ham emails. Using Bayesian filtering, he was able to provide a self-adapting method that keeps track of the evolution of spam emails [Gra04]. These filters commonly try to deduce how a dataset has been generated based on a set of training samples. This is done using a probabilistic model that embodies the generation assumptions by estimating the parameters of this model in a training phase. For the classification of new items, the Bayes' rule is applied to determine the class. Due to the naive assumption that all constituent parts of the item are statistically independent (although they are not), this approach is called Naive Bayes Classifier [McC98].

The main process of spam classification is conducted after the training phase as follows: First, a tokenizer is used to split the text of an email into its constituent parts called tokens. Once the tokenization is done, the classification step is carried out using this collection of tokens. The classifier identifies implicit relations between multiple textual descriptions. The classification consists of two main steps:

1. Calculation of the token/word frequencies.
2. Combination of these frequencies to the overall text frequency (statistical filtering).

Token Frequencies

To determine token frequencies, two methods are applied: In [Gra04], Paul Graham suggests regarding the tokens' occurrences in the single classes and calculating for each token w an affiliation probability $p(w)$ based upon the ratio of the single tokens' frequencies $b(w)$ for the "positive" (spam) and $g(w)$ for the "negative" (ham) class (cf. equation (6.1)). $p(w)$ is calculated according to Paul Grahams suggestions [Gra04] as follows:

$$p(w) = \frac{b(w)}{b(w) + g(w)} \tag{6.1}$$

with

$$b(w) = \frac{\#\ positive\ samples\ containig\ w}{\#\ positive\ samples}$$

and

$$g(w) = \frac{\#\ negative\ samples\ containig\ w}{\#\ negative\ samples}$$

This formula assumes an equal ratio of positive and negative samples. Paul Graham proposes scoring unknown tokens with a default probability of 0.4. In order to maintain the sample balance, we set the default probability to 0.5. As tokens of rare occurrence cannot be reliably categorized, using such tokens in the classification process might lead to misclassifications. In order to increase the tokens' confidence, Paul Graham suggests including only tokens occurring more than five times into further calculations. However, this extends the duration of the cold start phase (cf. paragraph "Challenges" in chapter 4) where insufficient data for a proper classification is available .

Gary Robinson addressed this shortcoming by extending the formula to reduce extreme results for tokens with little occurrences. He introduces the following smoothing parameter that expresses the assumed recommendation probability of an unknown token [Rob04]:

$$p'(w) = \frac{(sx) + (np(w))}{s + n} \tag{6.2}$$

where

- x is the assumed probability assigned to a token for which no (not enough) information is known based upon our general background information, e.g. the ratio of positive and negative samples or a neutral value for unbiased classification.
- s is the background information's strength. Increasing this variable also increases the sensitivity of the classifier with respect to changes within the user's habits.
- n is the number of samples containing the token w.
- $p(w)$ is calculated according to equation (6.1).

Default values in the spam filtering context are 0.5 and 1.0 for x and s respectively. This results in a score of 0.5 for unknown tokens. By slightly modifying the assumed probability, the current positive and its respective current negative sample ratio can be considered, at the price of biasing the classification process towards the favored category. This might lead to similar issues like those of Paul Graham's proposal with a default probability of 0.4.

Statistical Combination

To obtain an overall indication for a textual description $p(text)$, the single token frequencies have to be combined. This indication is modeled similarly to the concept of probability, varying from 0.0 (negative indication) to 1.0 (positive indication) with a neutral point at 0.5 (neutral indication). Bellow, we focus on the most significant statistical combination approaches:

- Graham Method: As a simplification, the single token probabilities $p(w_i)$ are assumed to be statistically independent and thus can be aggregated by multiplication:

  $$P(text) = \frac{P}{P + Q} \tag{6.3}$$

  with

  $$P = P(positive) = p(w_1)p(w_2)...p(w_n)$$

  and

  $$Q = P(negative) = (1 - p(w_1))(1 - p(w_2))...(1 - p(w_n))$$

- Robinson Geometric Means Method: To reduce the impact of extreme frequencies, Gary Robinson suggested combining the frequencies using their geometric means [Rob02]. Unlike Paul Graham's original approach that aims to reduce false positives by learning tokens from ham-messages twice [Gra03], this method leads to an indication for positive / negative class with the same level of sensitivity.

$$P = 1 - \sqrt[n]{(1-p(w_1))(1-p(w_2))...(1-p(w_n))}$$
$$Q = 1 - \sqrt[n]{p(w_1)p(w_2)...p(w_n)} \qquad (6.4)$$

- Robinson Fisher Method: With the Robinson Fisher method, Gary Robinson developed a more sophisticated way to ensure sensitivity for both recommendations and rejections. Consequently, the Robinson Fisher approach replaced the Geometric Means proposal. To formulate two null hypotheses one must assume ideal conditions, i.e. that token frequencies are pairwise independent, not uniformly distributed, and that the description consists of a random set of tokens. We then calculate a score based upon the hypotheses (”the message is a positive sample” / ”the message is a negative sample”) and using the inverse-chi approach of Fisher [Fis54].

$$H = C^{-1}(-2ln\prod_w p'(w), 2n) \qquad (6.5)$$

H denotes the ”hamminess” hypothesis (see equation (6.5)) with the inverse-chi-square function C^{-1} and the related degrees of freedom (twice the number of tokens) as input variables. As the inverse-chi-square approach is very sensitive to values near 0.0 (which is used as a negative indicator), a very similar hypothesis to H, S the hypothesis for "spamminess" is set up using the counter-probabilities $1 - p'(w)$ as follows:

$$S = C^{-1}(-2ln\prod_w (1-p'(w)), 2n) \qquad (6.6)$$

$$I = \frac{1 + H - S}{2} \qquad (6.7)$$

Equation (6.7) consolidates the two hypotheses S and H into a value ranging from 0.0 to 1.0, with values near to 0.5 signaling that there is as much indication for the positive as for the negative class. This results in a very high sensitivity to unsure classifications.

According to the spam filtering evaluation results of [Lou03a, Lou03b], the Robinson Inverse-Chi Bayesian Filter outperforms other implementations of Naive Bayesian Filters [Gra02] in terms of classification error rate.

Gary Robinson additionally introduces the so-called Effective Size Factor (ESF) to consider the lower informational value of spam mailings due to token redundancies. This factor modifies the degrees of freedom (second parameter of the Chi-Square function in equation (6.6)) for spam mailings. As it is specific to spam, the ESF is not used in our approach.

None of the outlined methods considers the real ratio of spam and ham. They all assume an equal amount of spam and ham by default. Therefore, the impact of a diverging distribution of spam and ham on the results must be considered carefully.

Classifier Modifications and Enhancements

In order to improve the classifier approaches mentioned in this section, various modifications - applicable on all presented types of classifiers - have been proposed over time as spam fighting research has progressed. These approaches mainly focus on the treatment of rarely occurring tokens, by including only the most important tokens in the statistical combination step, and by handling tokens that emerge multiple times in a single description.

The treatment of unknown or rarely occurring tokens, introduced by [Zdz05] as **hapaxes**, determines how the calculation step of related token frequencies is modified. The following options are possible:

- Ignore hapaxes by excluding them from the statistical combination step [Gra04].
- Apply the assumed recommendation probability by assigning a default probability to little known tokens (suggested by Paul Graham in [Gra04]).
- Use a smoothing approach by applying Robinson's formula containing a smoothing factor. The impact of this factor is reduced with the increasing token occurrences (see [Rob03]).

As pointed out in the tokenization step (see section 6.3.2), tokens vary according to their degree of relevance. Therefore, the **selection of decisive tokens** for statistical combination is also of major importance for the categorization precision. Common ways for incorporating tokens into the combination step are listed bellow:

- All tokens: Include all tokens regardless of their informational value (measured by the classification score).
- Window size: Include a fixed amount n of most interesting tokens. These tokens are obtained by measuring their distance from a defined neutral point (default 0.5). Only the n most distant tokens are taken into account.
- Radius: Include all interesting tokens i.e. tokens with frequencies beyond a specified distance around the neutral point. All tokens within this radius are considered to be of negligible impact on the overall score.

In the case of **multiple occurrences** of one token within a specific description, several weighting methods for including the token into the decision process are available:

- Single token inclusion: Each token is only incorporated one time in the combination step. Thus, tokens are treated independently from the number of times they appear in the same description.
- Single token inclusion with bonus for multiple occurrences: Each token is included only once within the combination step, but with a slight bonus (e.g. 0.001) for each further occurrence in the same description.
- Multiple token inclusion: Each token occurrence is included, even multiples, within the combination step

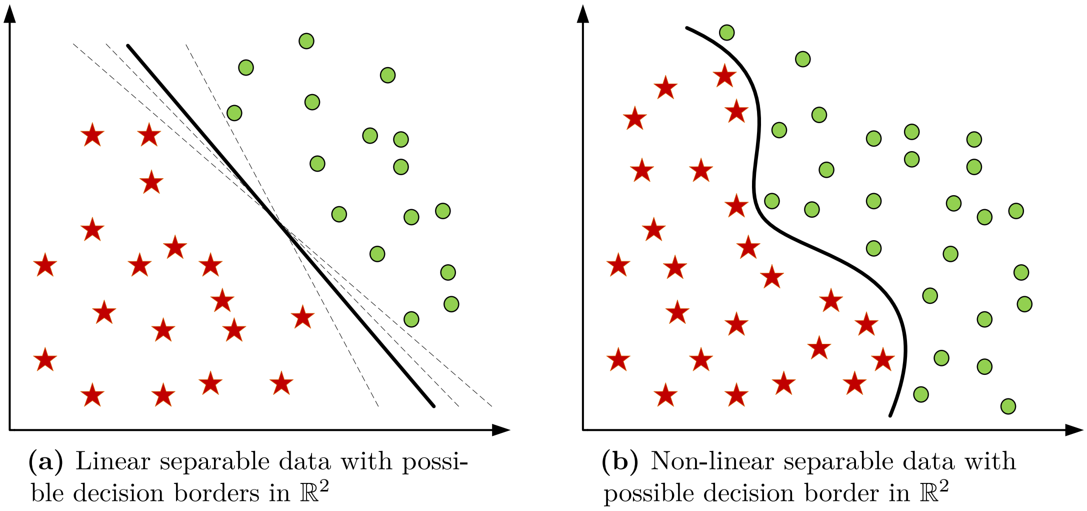
6.1.3 Support Vector Machines

Support Vector Machines (SVM) are often said to be among the most advanced classification mechanisms. They are widely used in several areas such as pattern recognition, information retrieval, image processing and retrieval, etc. The theoretical foundation of SVM goes back to Frank Rosenblatt, though the concept was introduced by Vapnik und Chervonenkis in 1974. The main principle behind a SVM is the determination of a decision border which is able to separate the input dataset according to its respective class. Commonly the dataset is composed of N samples $x_1 \dots x_n$ belonging to different classes as indicated by their class label y_i with $i \in [0,N]$. Therefore, the set of input data is represented by $D = \{(\vec{x}_i, y_i)\}$. Each sample in the dataset is represented by an vector forming a data point in $\mathbb{R}^m$. The process of determining the decision boarder can be roughly described as follows: Based on the dataset, the decision border from all possible borders (separators), which has the maximum distance from the data points of each class should be selected. Due to the distance maximization, the count of possible decision boarders (also called capacity of the classifier) is minimized. To conduct this process, each data point is extended by a small margin. By enlarging this margin, the capacity of the classifier is reduced in a stepwise manner until the optimal decision boarder remains. The capacity is closely related to the Vapnik Chervonenkis (VC) dimension further explained in [Bur98].

SVMs can roughly be categorized based on the characteristics of their datasets: the linear and the non-linear SVM.

Linear SVM - Hard Margin SVM

Linear SVM can be applied to two-class data ($y_i \in \{+1, -1\}$) which is linearly separable. In $\mathbb{R}^2$, this means that two sets of points can be completely separated by a line and in a n-dimensional space that at least one hyperplane can be found for separation. Figure 6.3 (a) shows an example 2D plot of linear separable data with several possible separators. In principal, any linear separator, as shown in the figure, fulfills the task of dividing the



(a) Linear separable data with possible decision borders in $\mathbb{R}^2$

(b) Non-linear separable data with possible decision border in $\mathbb{R}^2$

Figure 6.3: Linear vs. non-linear separable data.

given dataset (training data). For SVMs, this separator has to satisfy a simple criterion. It must maximize the distance between itself and the closest samples of each class. The bold decision boarder shown in figure 6.3 (a) satisfies this condition. In the following, this decision border will be referred to as a hyperplane or decision hyperplane. Figure 6.4 shows a detailed illustration of a linear SVM with its most important parameters. Those samples with the minimum distance to the decision hyperplane are commonly referred to as support vectors and the distance is called the margin. Generally speaking, the SVM tries to choose a hyperplane which maximizes this margin. Thus, the decision function of the SVM is fully specified by just a small subset of the input vectors, the support vectors. The decision hyperplane is defined as follows:

$$H = \vec{\omega}^T \vec{x} + b = 0 \tag{6.8}$$

This, with $\vec{\omega}$ as the normal vector of the hyperplane and b, the intercept term used for choosing a hyperplane among all hyperplanes predicular to $\vec{\omega}$. $\vec{\omega}$ is often referred to as the "weighting vector." Thus, according to [Man08], the decision function for classifying samples can be defined as follows:

$$f(\vec{x}) = sign(\vec{\omega}^T \vec{x} + b) \tag{6.9}$$

It classifies an m dimensional input vector into one of two classes $f_x : \mathbb{R}^m \mapsto \{+1, -1\}$. Thus, the hyperplanes $H1$ and $H2$ can be defined as $H1 : \vec{x}_i\vec{\omega} + b = 1$ and $H2 : \vec{x}_i\vec{\omega} + b = -1$ with a geometrical distance of $\frac{|1-b|}{||\vec{\omega}||}$ respectively $\frac{|-1-b|}{||\vec{\omega}||}$ to the origin. Hence, the distance d_+ to $H1$, and d_- to $H2$ respectively, and the separating hyperplane is given by:

$$d_+ = d_- = \frac{1}{||\vec{\omega}||} \tag{6.10}$$

Using d_+ and d_-, the geometric margin is defined as:

$$2m = \frac{2}{||\vec{\omega}||} = \frac{2}{\sqrt{\vec{\omega}^T\vec{\omega}}} \tag{6.11}$$

As mentioned previously, finding the optimal hyperplane requires the maximization of this margin. This task can be both formulated and transformed into an minimization problem, as follows:

$$max(\frac{2}{\sqrt{\vec{\omega}^T\vec{\omega}}}) \equiv min(\frac{\vec{\omega}^T\vec{\omega}}{2}) \tag{6.12}$$

with respect to the inequality constraint:

$$y_i(\vec{\omega}^T\vec{x} + b) \geq 1, \forall\{(\vec{x}_i, y_i)\} \in D \tag{6.13}$$

Thus, finding the hyperplane can be solved as a typical quadratic optimization problem where $||\vec{\omega}||^2$ is minimized. It is often done by formulating a dual optimization problem based on Lagrangian functions. This is due to its simpler handling of the problem and the option of allowing the generalization of the optimization step to non-linear SVMs [Bur98, Sch01]. Each constraint of equation (6.13) is substituted by an positive Lagrangian

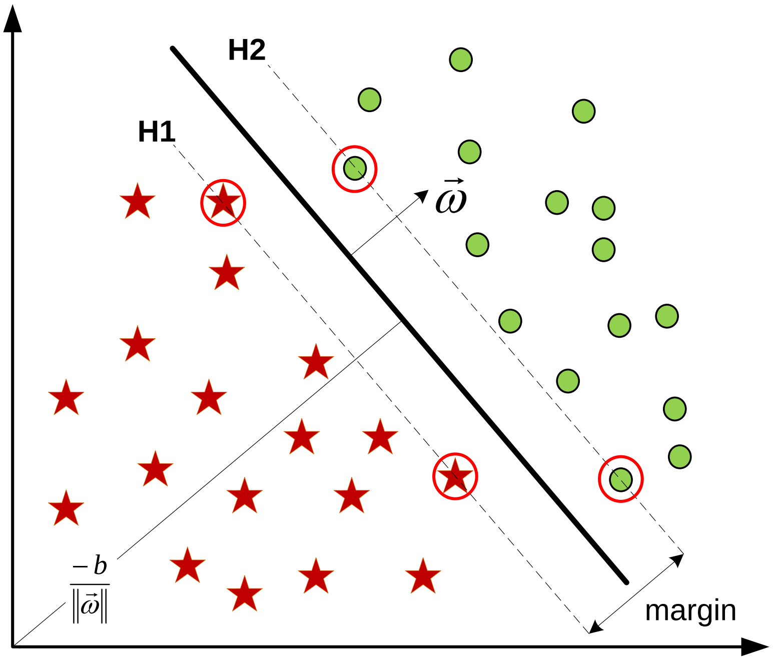


Figure 6.4: Linear SVM with its hyperplanes and support vectors [Bur98, adapted from figure 5].

multiplier α_i leading to the following Lagrangian form:

$$L \equiv \frac{1}{2}||\vec{\omega}||^2 - \sum_{i=1}^{N} \alpha_i(y_i(\vec{x}_i^T\vec{\omega} + b) - 1) \tag{6.14}$$

with

$$\vec{\omega} = \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i \tag{6.15}$$

and

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{6.16}$$

Thus, equation (6.14) is minimized according to ω and b, and maximized according to the Lagrangian multiplier α_i. For the formulation of the dual Lagrangian problem, we first expand the summands of equation (6.14) given by:

$$L_D \equiv \frac{1}{2}||\vec{\omega}||^2 - \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i^T \vec{\omega} - b\sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i \tag{6.17}$$

Using conditions (6.15) and (6.16) we can derive an equation without b and ω:

$$L_D \equiv \frac{1}{2}\sum_{i=1}^{N} \alpha_i y_i \vec{x}_i^T \cdot \sum_{j=1}^{N} \alpha_j y_j \vec{x}_j^T - \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i^T \cdot \sum_{j=1}^{N} \alpha_j y_j \vec{x}_j + \sum_{i=1}^{N} \alpha_i \tag{6.18}$$

Simplifying equation (6.18) the final for L_D can be derived:

$$L_D \equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \vec{x}_i^T \vec{x}_j \tag{6.19}$$

Using this equation, the optimal solution can be found by maximizing L_D. Please note that the final form depends solely on the training samples and the Lagrangian multiplier as the weighting vector ω, and the intercept parameter b are no longer present. Thus, by reducing the number of parameters, the complexity of the calculation is also reduced. For the classification step, the decision function (see equation (6.9)) is used with a small modification. ω is substituted by equation (6.15), leading to the following decision function:

$$f(\vec{x}_i) = sign(\sum_{i=1}^{N} \alpha_i y_i \vec{x}_i^T \vec{x} + b) \tag{6.20}$$

The whole classification process depends on the support vectors alone, where the Lagrangian multipliers α_i are greater than 0. All other samples are vanished by $\alpha_i = 0$ in equation (6.20).

Non-linear SVM - Hard Margin SVM

Thus far, we have focused on linear separable datasets, nevertheless this condition does not hold for many experimental datasets. Figure 6.3 (b) shows a typical example where a linear classifier is not applicable. The main idea behind non-linear SVMs is to map the data to a higher dimensional space where a linear separation is possible. This is commonly done using a transformation function $\phi : \vec{x} \mapsto \phi(\vec{x})$. An important property of this mapping is to not destroy the relatedness of samples in the dataset. In the field of SVMs, several efficient and feasible methods have been introduced, commonly referred to as a kernel, a kernel function or a kernel trick. Consequently, the scalar products in the training and in the test phase of the SVM are substituted by the kernel function K with $K(\vec{x}_i, \vec{x}_j) = \phi(\vec{x}_i) \cdot \phi(\vec{x}_j)$. Listed below are several popular kernel functions:

- Linear kernel: $K(\vec{x}_i,\vec{x}_j) = \vec{x}_i^T \vec{x}_j$
- Polynomial kernel: $K(\vec{x}_i,\vec{x}_j) = (1 + \vec{x}_i^T \vec{x}_j)^d$
- Radial basis function kernel:
  - $K(\vec{x}_i,\vec{x}_j) = exp(-\frac{(\vec{x}_i - \vec{x}_j)}{2\sigma^2})$ (Gaußkernel)
  - $K(\vec{x}_i,\vec{x}_j) = exp(-\gamma||\vec{x}_i - \vec{x}_j||^2), \gamma > 0$
- Sigmoid kernel: $K(\vec{x}_i,\vec{x}_j) = tanh(\gamma \vec{x}_i^T \vec{x}_j + r)$

With γ defining the slope of the function and r beeing an intercept constant.
By using the kernel, all other steps can be conducted as with linear separable data.

Soft Margin SVM

Often, experimental datasets have to be counted as non-separable due to erroneous or noisy data points. Especially in text-classification tasks where strange documents might lead to such situations this is certainly true [Man08, Section 15.2]. To cope with this situation, Cortes and Vapnik [Cor95] introduced the concept of soft margin SVMs. Soft margin SVMs extend the concept of SVMs to tolerate at least a small portion of erroneous samples and samples within the SVM's margin. In these cases, slack variables $\xi_i \geq 0$ are introduced for each sample of the dataset. For correct classified samples, the value of ξ_i is zero. If these samples are within the margin, the following condition holds: $0 < \xi_i \leq 1$. Samples within the margin but situated on the wrong side of the decision plane are labeled with $\xi_i > 1$. The minimization problem (cf. equation (6.12)) is modified as follows:

$$C \sum_{i=1}^{N} \xi_i + (\frac{\vec{\omega}^T \vec{\omega}}{2}) \tag{6.21}$$

The term C specifies the costs related to a violation of the hyperplanes and the decision border. It can be seen as a mechanism for controlling the trade-off between the complexity of the decision border and the frequency of errors. Note that a very low value of C leads to a broad margin, leading to a good generalization capability but a higher possibility of misclassifications. However, for a value of C going to infinity, the soft margin SVM degenerates to a standard SVM where no samples within the margin are allowed. For the Lagrangian problem formulation, the constraints of the equation are also modified in respect to C. The equation is as follows:

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{6.22}$$

with

$$0 \leq \alpha_i \leq C \tag{6.23}$$

Thus, the Lagrangian multipliers are limited by C.

6.1.4 Latent Semantic Indexing

Latent Semantic Indexing (LSI), also called Latent Semantic Analysis (LSA), was introduced in 1990 by Deerwester et al. [Dee90]. It is an approach for discovering latent semantic relations between words, hidden by the variability of word choice, in their specific contexts based on their statistical properties. Its main aim was to solve or at least ease the problem faced by most information retrieval methods referred to as the vocabulary problem [Fur87]. It is a common problem in human-computer interactions and describes the discrepancy between the words different humans use for a meaning (thing), and the words used to represent a meaning in a system. Well known examples are synonyms, where different words refer to the same or to a very similar meaning, and polysemes, where the same word is used to refer to multiple meanings. LSI is said, at least to a certain extend, to reveal the semantic relations between words and therefore facilitate the handling of this problem. The main process of LSI can be described as a typical low-rank approximation of a term-document matrix C of dimensions $M \times N$ [Man08]. This matrix is defined as follows:

$$C = \begin{pmatrix} m_{11} & m_{12} & \dots & m_{1N} \\ m_{21} & m_{22} & \dots & m_{2N} \\ . & . & . & . \\ . & . & . & . \\ m_{M1} & m_{M2} & \dots & m_{MN} \end{pmatrix} \tag{6.24}$$

where documents are added as rows and terms as columns. Each entry m_{ij} indicates the occurrence of term i in document j. This indication is often modeled as a term frequency (tf) or term frequency inverse document frequency (tf-idf) (cf. section 6.3.6). Thus, even for small datasets, C is highly dimensional with several thousand terms and documents listed in the matrix.

The approximation problem can be formulated as follows: For the term-document matrix C we try to find a Matrix C_k with $rank(C_k) = k$ and $rank(C_k) << rank(C)$ with a minimum difference between C and C_k, considering the Frobenius norm $X = C - C_k$ defined as:

$$||X||_F = \sqrt{\sum_{i=1}^{M} \sum_{j=1}^{N} x_{ij}^2} \tag{6.25}$$

The rank of a matrix is defined by the number of linearly independent rows or columns in the matrix. This minimization problem is solved based upon the mathematical foundation of the Singular Value Decomposition (SVD). The whole process is conducted in the following two steps:

1. **SVD**
   SVD is often referred to as the process of decompounding a given $M \times N$-Matrix C into the product of submatrices $\mathbb{U}$,Σ and V derived from C's eigenvectors:

   $$C = \mathbb{U} \Sigma V^T \tag{6.26}$$

   where

   - $\mathbb{U}$ is a $M \times M$ matrix with the eigenvectors of CC^T (left eigenvectors). $\mathbb{U}$ represents the terms with u_{mn} representing the intersection between the co-occurrence of term m and term n in the collection of documents.
   - Σ is a $M \times N$ diagonal matrix whose diagonal entries are the singular values of C in a decreasing order. The singular values are calculated as the square root of the eigenvalues. By convention, all values except the diagonal values are omitted, building the reduced SVD.
   - V is a $N \times N$ matrix with the eigenvectors of C^TC (right eigenvectors). Therefore, V represents the overlapping between different documents.

   Figure 6.5 shows the matrix decomposition of C. In the upper part the decomposition for a matrix where $M > N$ and in the lower part where $N > M$ is shown.

2. **Rank Approximation**
   Figure 6.6 illustrates the reduction of a matrix to rank k. Due to the construction of Σ, the low rank approximation to a specific rank of r can be done by substituting all entries with an index greater than r by zero. Thus, as the singular values in Σ

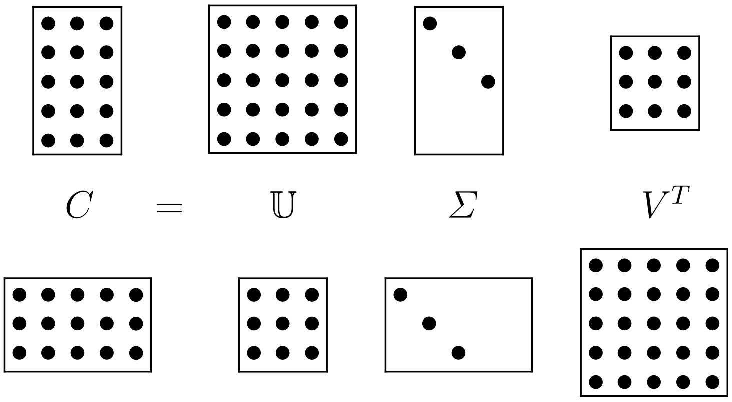


Figure 6.5: Exemplary Singular Value Decomposition (SVD) of a matrix.

are in a decreasing order, values with the smallest contribution to the overall matrix product are skipped [Man08]. In fact, skipping single very small values will not greatly alter the product. It is certain that, if r is equal to the value of $rank(C)$, the difference between C and C_k will be 0, whereas with decreasing k latent relations in the dataset are amplified and emphasized. The determination of k must be done according to the concrete area of application. In text classification, k is often given a relatively small value (e.g. 100) which typically gives the best results [Ber95].

LSI Queries

LSI queries are used to determine the similarity between a query and the term-document matrix. These queries are formulated as term vectors $\vec{q}$ of dimension N where all terms occurring in the query (query document) and in the index are represented by 1, and all others by 0. A common method of measuring the similarity between a document $\vec{d}$ and the query vector $\vec{q}$ in the vector space model [Sal86] is given by the cosine similarity:

$$sim(\vec{q}, \vec{d}) = \frac{\vec{q}\vec{d}}{|\vec{q}||\vec{d}|} \tag{6.27}$$

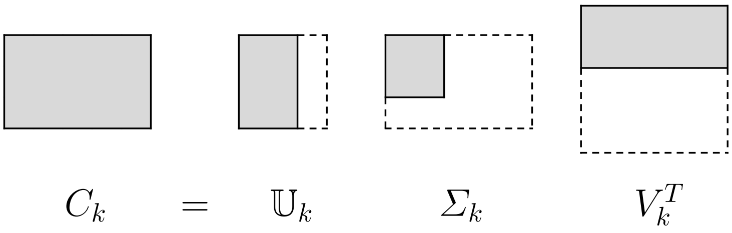


Figure 6.6: Exemplary low-rank approximation of a matrix.

To benefit from the dimension reduction, the query vector must also be transformed according to k. Therefore $\vec{q}$ is represented in the LSI space as follows:

$$\vec{q}_k = \Sigma_k^{-1} \mathbb{U}_k^T \vec{q} \tag{6.28}$$

Depending on the content of the query and based upon a similarity measure, the distance between two documents, two terms or between a typical query and a document can be measured. Although the approximation of the model is not equal to the original model, it is said to preserve the main relations between documents and queries within the similarity measure.

All mechanisms mentioned in this section (section 6.1) have been implemented in our system. Required adaptations of each classification mechanism for their application in the realm of TV are covered in section 6.3.6. For a comparative evaluation of the different mechanism, the reader may refer to section 6.5. The following section details the system and its components.

6.2 System Overview

As depicted in figure 6.7, our system is divided into 5 main parts: the consumer side with the personal remote control (see section 6.2.1), the data and communication layer (see section 6.2.3), the recommendation system (see section 6.2.5), the media center environment (see section 6.2.2) and the metadata layer (see section 6.2.4) with its different sources for EPG data. The system has mainly been implemented in Java to support mobile devices and to facilitate platform independence. Multi-user support is enabled by the assignment of unique identifiers to each personal device. Optionally, a simple login form can be used for user identification and adequate session tracking. Thus, the system is capable of tracking interaction with different users and personalized recommendation generation based upon their individual user profiles.

In considering the startup phase of the system, several process steps are conducted. The first step is the connection setup of the user's personal device. All connection to client devices are handled by the data and communication layer. Once the connection is established, the whole interaction (also called click stream) such as common remote control commands (e.g. channel up/down), search and program requests or program tag annotations are forwarded to the communication manager. The user commands are handed over to the media center environment for the execution and are also stored in the usage history database. Simple remote control commands are processed and executed directly by the media center. Afterwards, queries concerning program information, such as querying the prime time program of the day, are handled by the metadata layer and components of the recommendation system. Recommendations are generated in both the collaborative and the content-based recommendation component by combining the usage history with the related program descriptions. A crucial part of our system is the interpretation of user actions, see section 6.3.1 and table 6.3 for a discussion of this interpretation.

Then, the program descriptions are imported from the DVB stream, via the Video Disk Recorder (VDR) and from the internet using the data provider EPGdata.com (see

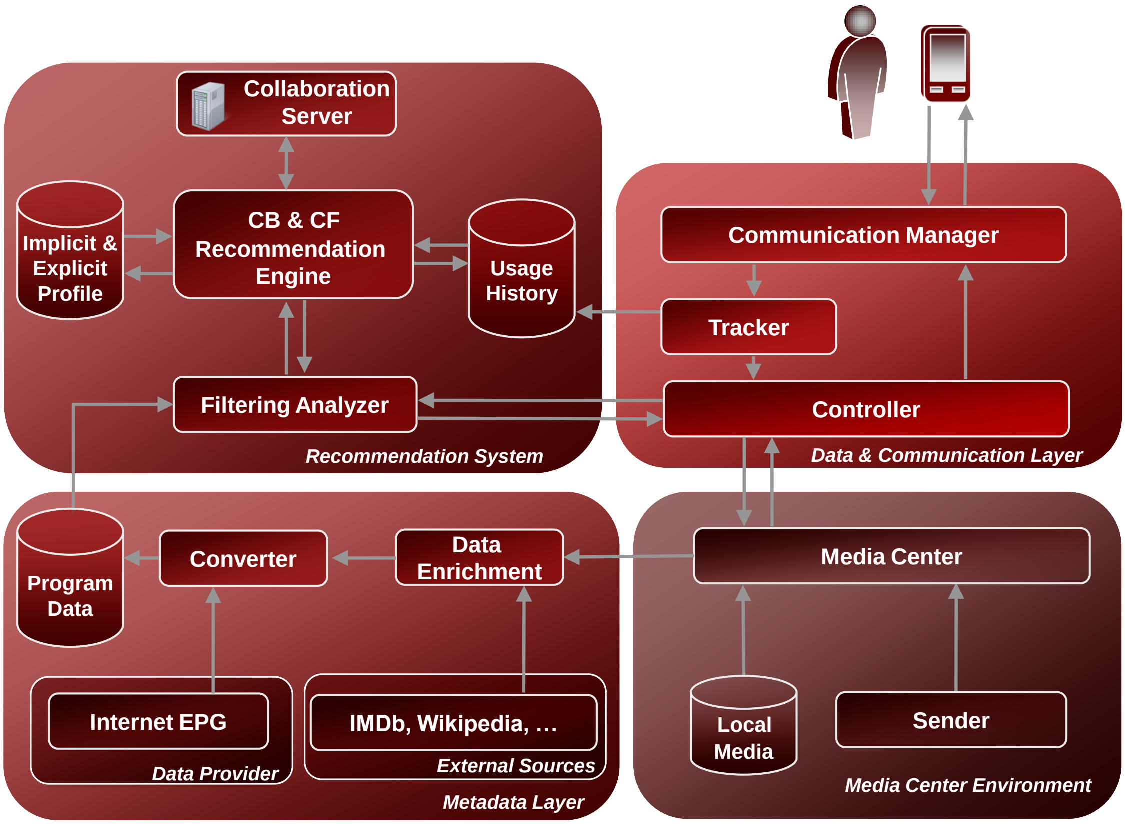


Figure 6.7: System overview of personalTV.

section 2.2 for a discussion on metadata formats). Finally, after the conversion into a uniform standardized program data representation (currently TV-Anytime and XMLTV are supported), which merges the data providers' common program metadata elements, the data is stored in a program database. Due to this generic internal representation, the process may be easily adapted to arbitrary data formats.

In the next sections the system's main components are presented in detail.

6.2.1 Personal Remote Control

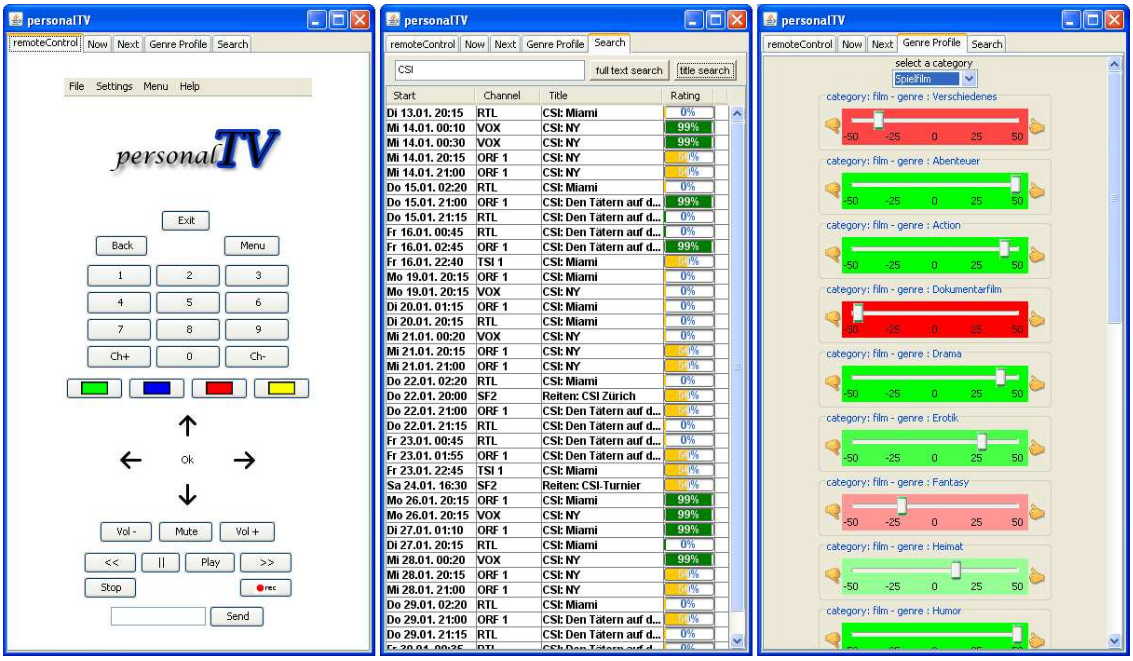
In our approach, a personal device such as a PDA or Smartphone is used to control the system. Each personal device is assumed to belong to a unique user. Thus, this device can be used as a source for identifying the user. Due to advanced capabilities of such devices compared to ordinary remote controls, new means of building the control interface and the introduction of advanced interaction options are made possible. Our personal remote control has been designed for use on touch-screen devices. It offers a typical remote-control interface, well known from the layout of most TV remote controls. In addition to the standard functionality, our remote control provides adaptable program listings with detailed program information, program rankings according to the viewer's profile, a search function and the possibility of adding tags to the programs. The device can also be used to control the system's behavior in the absence of the user, e.g. by setting thresholds for certain actions as reminders, and an auto recording function for interesting programs. Additionally, explicit user preferences can be set manually. Figure 6.8 illustrates our personal remote control with several screenshots.

6.2.2 Media Center Environment

To provide a media center environment, this work uses the popular open-source program VDR[1]. The VDR enables a Linux PC to function as a digital receiver and video recorder. It supplies the Simple VDR Protocol (SVDRP) to send commands to the media center over a plain TCP connection. Using this protocol, EPG data from the VDR, as well as information about channel assignment, timers and recorded programs can be retrieved. In general, the support of any media center software with an open control interface for the communication with the remote control can be used in our system.

6.2.3 Data and Communication Layer

The data and communication layer consists of the communication manager and the controller. The communication manager receives the user's commands from the remote control and forwards them to the media center, as well as to the usage history database. The commands are sent by the controller to the VDR. Further, responses of the VDR are forwarded to the remote control. VDR's SVDRP interface is used for communication. Additionally, the controller keeps track of the mapping between the VDR's Channel



select a category: Spielfilm

CSI

Start	Channel	Title	Rating
Di 13.01. 20:15	RTL	CSI: Miami	0%
Mi 14.01. 00:10	VOX	CSI: NY	99%
Mi 14.01. 00:30	VOX	CSI: NY	99%
Mi 14.01. 20:15	ORF 1	CSI: NY	50%
Mi 14.01. 21:00	ORF 1	CSI: NY	50%
Do 15.01. 02:20	RTL	CSI: Miami	0%
Do 15.01. 21:00	ORF 1	CSI: Den Tätern auf d...	99%
Do 15.01. 21:15	RTL	CSI: Den Tätern auf d...	0%
Fr 16.01. 00:45	RTL	CSI: Den Tätern auf d...	0%
Fr 16.01. 02:45	ORF 1	CSI: Den Tätern auf d...	99%
Fr 16.01. 22:40	TSI 1	CSI: Miami	50%
Mo 19.01. 20:15	ORF 1	CSI: Miami	0%
Mo 19.01. 20:15	VOX	CSI: NY	0%
Di 20.01. 01:15	ORF 1	CSI: Miami	0%
Di 20.01. 20:15	RTL	CSI: Miami	0%
Mi 21.01. 00:20	VOX	CSI: NY	0%
Mi 21.01. 20:15	ORF 1	CSI: NY	50%
Mi 21.01. 21:00	ORF 1	CSI: NY	50%
Do 22.01. 02:20	RTL	CSI: Miami	0%
Do 22.01. 20:00	SF2	Reiten: CSI Zürich	50%
Do 22.01. 21:00	ORF 1	CSI: Den Tätern auf d...	50%
Do 22.01. 21:15	RTL	CSI: Den Tätern auf d...	0%
Fr 23.01. 00:45	RTL	CSI: Den Tätern auf d...	0%
Fr 23.01. 01:55	ORF 1	CSI: Den Tätern auf d...	50%
Fr 23.01. 22:45	TSI 1	CSI: Miami	50%
Sa 24.01. 16:30	SF2	Reiten: CSI-Turnier	50%
Mo 26.01. 20:15	ORF 1	CSI: Miami	99%
Mo 26.01. 20:15	VOX	CSI: NY	99%
Di 27.01. 01:10	ORF 1	CSI: Miami	99%
Di 27.01. 20:15	RTL	CSI: Miami	0%
Mi 28.01. 00:20	VOX	CSI: NY	99%
Mi 28.01. 20:15	ORF 1	CSI: NY	50%
Mi 28.01. 21:00	ORF 1	CSI: NY	50%
Do 29.01. 02:20	RTL	CSI: Miami	0%
Do 29.01. 21:00	ORF 1	CSI: Den Tätern auf d...	50%
Do 29.01. 21:15	RTL	CSI: Den Tätern auf d...	0%
Fr 30.01. 00:35	RTL	CSI: Den Tätern auf d...	0%

Figure 6.8: Screenshots of the remote control, search dialog and explicit profile definition dialog.

[1] Video Disk Recorder - http://www.cadsoft.de/vdr/

settings and the personal remote control's properties as well as of the synchronization of the personal remote control's commands with the VDR commands. In case of an error, the media center's feedback messages are processed and forwarded again. The action logs of the users as well as their settings, program information and classifier specific data are stored in a database, currently a PostgreSQL database. In the startup phase of the system, the Controller updates information about recorded programs and program timers kept by the VDR. The periodical update of the program information in our database is initiated here as well.

6.2.4 Metadata Layer

The Metadata Layer's main purposes are the gathering and the transformation of EPG data. Currently, two means for gathering metadata are available:

1. EPG data within the DVB: As far as digital television is concerned, by default the VDR uses EPG data which are transmitted via the DVB stream using DVB-SI specification (cf. section 2.2.1). The acquired EPG data are stored in an internal program database. Depending on the broadcasting station, VDR provides EPG information on start time, channel, length, title, subtitle, aspect ratio, audio and a description of the program. Program data is available up to 7 days in advance. Depending on the channel, the basic EPG data show deficits and gaps. Some broadcasters only transmit "now & next" program metadata containing only default data like start time, length and title of the program.

2. Internet EPG: Several providers offer EPG data in the form of downloadable packages on the internet. These professionally prepared metadata provide a very comprehensive source for program descriptions. Currently, we use EPG data from epgData.com, due to the extensive and high quality nature of it's information. It offers information such as synopsis, genre, involved persons, parental guidance and theme. Although many open and standardized EPG data formats exist, most Internet EPG providers, including epgData.com, use their own proprietary format.

After collecting EPG data, the further processing steps depend heavily on the data's origin. In the case of professional data received from Internet EPG's, only the conversion step is done. In this step, data is converted to the TV-Anytime format using the API provided by the BBC[1]. Using this format, it is easy to support further data sources. Moreover, it guaranties interoperability with other TVA-based TV systems such as Ifanzy [Akk06], Avatar [BF07] or the personalization system of Weiss et al. [Wei08]. In the case of missing features, such as genre or persons involved, a data enrichment step is needed to guarantee an adequate amount and quality of metadata. This step is necessary if EPG data is collected from the DVB stream. The main reason for this is, that missing metadata negatively impacts the performance of the entire recommendation generation process (cf. section 6.2.5). Within this process, user preferences are derived based upon the relation between user actions and related metadata. Hence, for superior performance more advanced metadata are required.

[1] BBC TV-Anytime API – http://www.bbc.co.uk/opensource/projects/tv_anytime_api/

Enrichment

The metadata enrichment component enables the enhancement of rudimentary metadata delivered within the DVB stream by feature rich descriptions. Figure 6.9 shows a rough overview of this system component. The whole process starts with the EPG data provided by our media center component, the VDR. EPG data is gathered by continuously scanning the current channel. When the media center is in idle mode, a full EPG scan is frequently initiated to gather EPG data for all available channels. This is done by flipping through the entire channel list and scanning each individual channel for EPG data. For a system, with multiple tuner setup, this is done as a background process which leads to more up-to-date EPG information. Program metadata is saved in a proprietary format (cf. VDR EPG.data Format[1]). After conversion to the internal representation, data is ready for enrichment. This enrichment step is managed by the parser controller. This component integrates all available parsers and controls the individual enrichment steps. The metadata enrichment is done by using several web resources which make publicly available information on TV programs. Table 6.2 shows a rough comparison of the data elements available from each data source. As the data elements may not always be filled for each program, redundancy between the sources can be used to gather all information needed for a comprehensive program description.

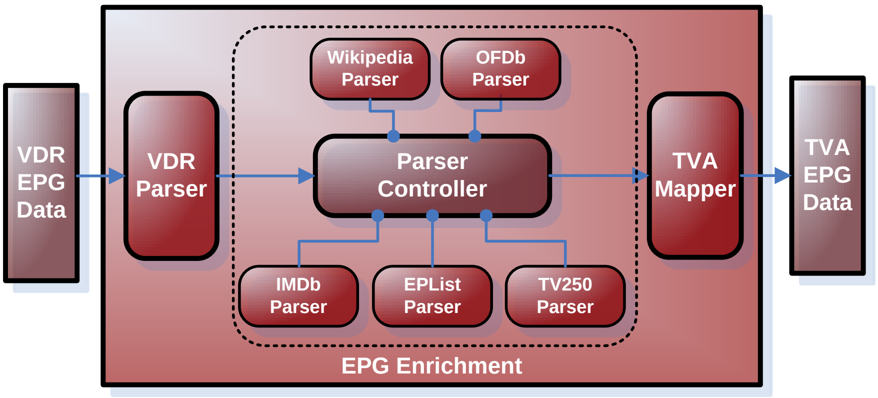


Figure 6.9: Architecture of the EPG data enrichment component.

For each data source, a separate metadata parser, controlled by our Parser Controller, has been implemented:

- EPList parser: EPList is specialized for TV series and offers more than 470 episode- and season lists[2]. The information is very limited, only the episode titles, season titles and numbers are available. Parsing can be done based upon the web pages of the web portal or a simple text file available for download. The metadata format of EPList is defined as plaintext and very easy to parse. The content of EPList is maintained and supplemented by a small number of registered users. Thus, the quality and consistency of the data is directly dependent on this user group.

[1] VDR EPG.data – http://www.vdr-wiki.de/wiki/index.php/Epg.data

[2] Accessed on November 19, 2010 – https://ssl.constabel-it.de/eplists.constabel.net/

- IMDb parser: The Internet Movie Database (IMDb) is one of the most comprehensive sources for movie and TV metadata, ranging from title to the full cast information. It offers more than 1,7 million[1] descriptions. As parsing of the IMDb portal is forbidden, we use a local copy of the movie database for retrieval. Database dumps are publicly available and frequently updated. IMDb gathers information directly from the production side. Additionally, user may enter and edit data. For user generated content predefined input forms are offered. Furthermore, the consistency and quality of the data is directly checked by the IMDb team. The level of available data is consistently very high. Due to the update and quality assurance process of IMDb, information about a new title takes between 6 and 8 weeks until it is available.

- OFDb parser: The Online-Filmdatenbank (OFDb) is the biggest German movie database and offers more than 190.000 titles[2]. It offers metadata in a very similar way to IMDb. The input form for new data also follows the style of the IMDb input form. Program metadata is retrieved via the search functionality of the website and a parsing step. New entries to the database are validated by the OFDb team.

- TV 250 parser: TV 250[3] is closely related to IMDb. It offers TV schedule listings for IMDb's top 250 movies. Besides basic program information of IMDb, it mentions the channels, start times and dates. Additionally an indicator is offered to point out programs that are ad-free. Information of TV 250 is provided by a small community. For parsing, the website's TV program list is directly accessed and processed.

- Wikipedia parser: Articles about movies, TV shows and series are entered in Wikipedia based upon predefined templates[4]. These templates can be seen as a kind of metadata format specification. Using this specification, the parser is able to process all entered information. Because of the community based quality assurance mechanism most metadata descriptions can be considered as valid and of high quality. Furthermore, information about recently published content is often available. Nevertheless, the quantity of information strongly varies depending on the authors of the metadata description.

The parser controller monitors the initialization of each individual parser and aggregates the returned program information. Due to its open structure, additional sources such as OMDb[5] or MovieLens[6] and the appendant parser can be integrated easily. In a qualitative comparison of the different data sources accessed by our system, the following order among the data sources has been determined:

1. IMDb: Due to the huge amount of available high quality data, IMDb can be considered the most important source.

[1] Accessed on November 19, 2010 – http://www.imdb.com/stats

[2] Accessed on November 19, 2010 – http://www.ofdb.de/view.php?page=stats

[3] TV 250 – http://www.tv250.de/

[4] Wikipedia TV and movie templates German – http://de.wikipedia.org/wiki/Kategorie:Vorlage:Infobox_Film_und_Fernsehen

[5] Open Media Database – http://www.omdb.org

[6] MovieLens – http://www.movielens.org

	EPList	IMDb	OFDb	TV 250	Wikipedia
Title (original / german)	original	both	both	german	both
Credits information	–	full cast info	actors, directors, rolenames	–	actors, directors, rolenames
Genre	–	√	√	–	√
Episode- / Seasoninformation	Episode-, Seasonlist and -count	Episode-, Seasonlist and -count	–	–	Episode-, Seasonlist and -count
Synopsis	–	English	German	–	Multilingual
Add free	–	–	–	√	–
Keywords	–	√	–	–	–
Moderator	–	√	–	–	√
Country and date of production	–	√	√	–	√
Concept	–	√	–	–	√
Rating	–	√	√	–	–
Avg. rating	–	√	√	√	–
Vote count	–	√	√	–	–
Original length	–	√	–	–	√
Age rating	–	√	–	–	FSK

Table 6.2: Available metadata elements on different sources.

2. Wikipedia: It offers the latest information. Information for live events can also often be found on Wikipedia. However, the data quality on this site is only ensured by the community.

3. OFDb: OFDb has been considered the third most important because of its high similarity to IMDb though the amount of available data is much smaller.

4. TV 250 and EPList: These two sources can be considered to be equally important, as they both provide very specialized information for a small number of TV programs.

We are certainly not able to guarantee that the right TV program will always be found and that the right program metadata are added. Especially movies like ”Godzilla,” which have been filmed several times over the years, are hard to distinguish based solely upon the rudimentary data delivered within the DVB stream, the VDR EPG data. Another problem is the enrichment of news programs, live events and other programs with a close relation to current events, where hardly any information can be retrieved. In such cases, only the VDR EPG data are used as a kind of fall back program annotation. Nevertheless, in the case of more than 90 percent of the programs viewed by our users, we are able to find additional information. Especially for movies, this information exceeds even the program metadata gathered from Internet EPGs in terms of amount and quality.

After the parsing steps are complete, the program descriptions are converted into a TV-Anytime representation that is compliant with the standard. For representing, converting and processing TV-Anytime documents, the TV-Anytime Java API from BBC[1] has been used. Please note that the tags/keywords offered by IMDb can be easily used as basic tag annotation for our collaborative media recommender component, as described in section 6.4.

In the following, a sample output of the enrichment process is shown. Listing 6.3 shows the VDR event description of the movie ”The Rock - Fels der Entscheidung” extracted from the DVB stream. *C* is used to describe the channel, here ”ProSieben on Astra 19.2 East.” *E* describes the start time / date (Fri Feb 12 20:15:00) and the duration (140 min) of the broadcast event. The title is mentioned after descriptor *T* and the synopsis after descriptor *D*. Descriptors $X1$ and $X2$ provide information on video and audio, such as aspect ratio, audio streams and languages. For a detailed description of the VDR EPG data format, interested readers may refer to the VDR wiki[2].

Listing 6.3: VDR EPG data example

```
C S19.2E-1-1107-17501 ProSieben
E 31083 1266002040 8340 50 5
T The Rock - Fels der Entscheidung
D General Hummel und seine Elitetruppe haben die Gefängnisinsel Alcatraz ...
X 1 01 deu 4:3
X 2 03 deu deutsch
X 2 05 deu Dolby Digital 2.0
```

Listing 6.4 shows an excerpt of the TV-Anytime output of the enrichment process. In addition to the title information of the VDR event, the original movie title has also been extracted. A detailed German synopsis has been extracted from OFDb, further an English synopsis would have been available from IMDb. Next, a set of keywords is listed. A genre annotation is also commonly available for most broadcast events, in this example ”Action” and ”Thriller.” The *CreditList* offers information on actors, directors, producers, moderators and many more. Additionally, actors are annotated with their role name, if available. Moreover, ratings from sources like IMDb and OFDb, listed next, can be taken into account as a first indicator of the popularity of a TV program. As shown in the listing, the event description can be extended by the production country, the production year, a FSK age rating and an indicator for ad freeness. Broadcast information is directly taken from the VDR event description.

Listing 6.4: EPG enrichment output

```
<ProgramInformation>
  <BasicDescription>
    <Title type="main"><![CDATA[The Rock - Fels der Entscheidung]]></Title>
    <Title type="original"><![CDATA[The Rock]]></Title>
    <Synopsis length="medium"><![CDATA[Ein Terrorkommando unter Ex-General Hummel
        Ein Terrorkommando unter Ex-General Hummel besetzt im ...]]></Synopsis>
    <Keyword type="main"><![CDATA[alcatraz]]></Keyword>
    <Keyword type="main"><![CDATA[prison]]></Keyword>
    <Keyword type="main"><![CDATA[nerve-gas]]></Keyword>
```

[1] BBC-TV-Anytime API – http://www.bbc.co.uk/opensource/projects/tv_anytime_api/

[2] VDR EPG.data – http://www.vdr-wiki.de/wiki/index.php/Epg.data

```
    ...
    <Genre href="urn:tva:metadata:cs:IntentionCS:2002:1" type="main">
      <Name><![CDATA[Action]]></Name>
    </Genre>
    <Genre href="urn:tva:metadata:cs:ContentCS:2002:3" type="main">
      <Name><![CDATA[Thriller]]></Name>
    </Genre>
    <CreditsList>
      <CreditsItem role="urn:mpeg:mpeg7:cs:MPEG7RoleCS:Actor">
        <PersonName>
          <mpeg7:GivenName>Sean</mpeg7:GivenName>
          <mpeg7:FamilyName>Connery</mpeg7:FamilyName>
        </PersonName>
        <Character>
          <mpeg7:GivenName>John Patrick</mpeg7:GivenName>
          <mpeg7:FamilyName>Mason</mpeg7:FamilyName>
        </Character>
      </CreditsItem>
      ...
      <CreditsItem role="urn:mpeg:mpeg7:cs:MPEG7RoleCS:Regie">
        <PersonName>
          <mpeg7:GivenName>Michael</mpeg7:GivenName>
          <mpeg7:FamilyName>Bay</mpeg7:FamilyName>
        </PersonName>
      </CreditsItem>
      <CreditsItem role="urn:mpeg:mpeg7:cs:MPEG7RoleCS:Idea">
        <PersonName>
          <mpeg7:GivenName>Douglas</mpeg7:GivenName>
          <mpeg7:FamilyName>Cook</mpeg7:FamilyName>
        </PersonName>
      </CreditsItem>
      ...
    </CreditsList>
    <Ratings>
      <Rating>
        <System>OFDB</System>
        <Value>8.26/10</Value>
        <Votes>2465</Votes>
      </Rating>
      ...
    </Ratings>
    <Country>![CDATA[Vereinigte Staaten]]</Country>
    <Addfree>false</Addfree>
    <Year>1996</Year>
    <FSK>18</FSK>
  </BasicDescription>
  ...
  <ServiceInformation serviceId="ProSieben">
    <Name>ProSieben</Name>
    ...
    <PublishedStartTime>2010-02-12T20:15:00Z</PublishedStartTime>
    <PublishedDuration>PT2H20M0S</PublishedDuration>
...
</ProgramLocationTable>
```

For a detailed discussion and a comparison of different TV metadata standards interested readers may refer to section 2.2. This section also details the reasons for choosing TVA as the basic EPG data standard in our systems.

6.2.5 Recommendation System and Collaboration Server

Recommendation engines have become an important part of many systems. Thus, a large amount of different approaches have been proposed. Commonly these methods are categorized as *content-based*, *collaborative* and *hybrid* approaches. In the following section we briefly mention the main filtering and classification concepts used in our system. For a comprehensive survey of different recommender systems and detailed categorization, the interested readers may refer to chapter 4 and to the paper of Adomavicius and Tuzhilin [Ado05].

The *Recommendation System* with its *Collaboration Server* is one of the main components of our system. It utilizes the user's viewing history, composed by the click stream of individual users, and combines it with program information provided by the *Metadata Layer*. On the methodical level this component can be partitioned into a solely content-based (discussed in section 6.3) and into a collaborative part (discussed in section 6.4). Figure 6.10 shows a rough overview of the different classification methods, based upon Spam Filter, Support Vector Machine and Latent Semantic Indexing, and a collaborative filtering approach, used in our system. For a detailed discussion of the fundamental concepts of these methods, the reader may refer to the beginning of this chapter (section 6.1) and for a discussion of the current application in our TV recommendation system, to section 6.3.6 and to section 6.4.2. As a central component and a connector between the content-based and the collaborative system part, the ”Classifier Ensemble” is used. It is a composition of multiple classifiers and makes use of our dynamic weight adaptations approach (cf. section 6.5.5) for the individual classifiers. A detailed explanation of this component can be found in section 6.3.3. Recommendations are generated on both the content-based and the collaborative, sides of the system and forwarded to the user. Depending on the user's preferences, recommendation scores can be merged or differently ranked for the overview of his or her program rankings and recommendations.

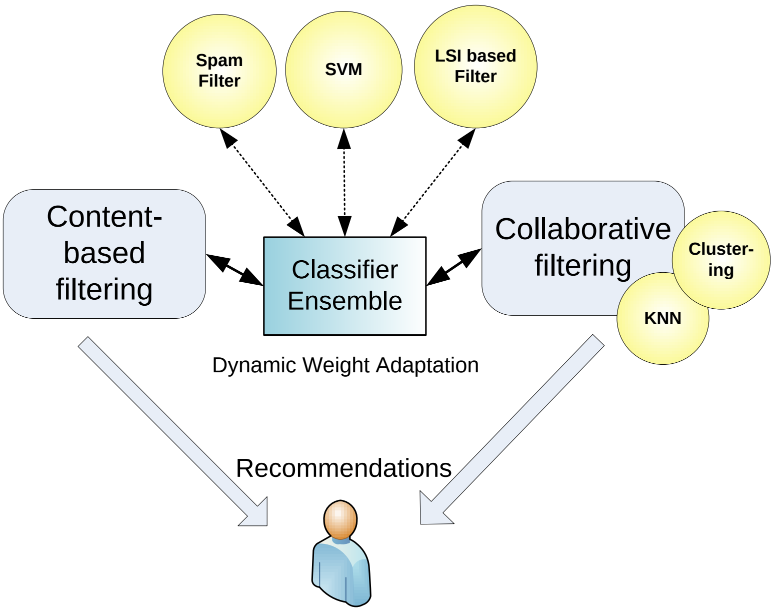


Figure 6.10: Methodical overview of the media recommender framework.

6.2.6 Time Context of User Profile and Interest Drift

Time is an important factor in recommendation systems for several reasons. The shifting interests of users is one of the most obvious points. Most authors, such as Koychev et al. [Koy00], Yu et al. [Yu08] and Cremonesi et al. [Cre10], agree that user's interests change over time and therefore profiles have to be adapted to represent his or her current preferences. Moreover, most recent observations and user actions are said to represent the current ”interest” better than older ones. Thus, adequate update and adaptation strategies are needed. Another issue is the memory requirement of a profile. As a user profile expand over time, adequate purge policies are needed to keep the amount of profile data in a manageable size. In resource-constrained environments such as set-top boxes this policy becomes even more important.

For coping with both issues, the most common method is the time window model. This model only considers observations within a specific time window, using them for the training of the user model. As an improvement on this mechanism, the window size can be adapted dynamically according to the current accuracy of the system.

Another method is the use of different user models. A short-term model, keeping track on recent observations, and a long-term model, responsible for older and more stable user interests. To make predictions, first the shot-term model is used, and if no accurate classification is generated, then the long-term model is used.

In [Koy00] a gradual forgetting function $w = f(t)$ is defined to provide weights for each observation in relation to the observations time. $f(t)$ is modeled as a linear function for a window of n observations in the following way: $w_i = -\frac{2k}{n-1}(i-1) + 1 + k$ with $k \in [0,1]$ and i being an observation counter $i \in [1,n]$.

For reducing the number of observations taken into account mechanisms from the spam fighting domain and different page replacement algorithms can be used. For instance, observations with a low occurrence count in the profile that had become stale after a given period of time can be removed. Moreover, strategies like Least Recently Used (LRU) or LRU-2 are also applicable for purging user profiles.

In our system we suggest the use of a kind of ”logarithmic” grading of user models. A current short-term user model is kept in a window model approach and different long-term models are used for increased window sizes. For instance, a short-term model is constructed for the last month and long-term models for the last 6 month, the last year and so on. To keep the amount of data small enough for the long-term models to handle, only the most frequent observations are used. Interest prediction can be made as a weighted combination of predictions coming from short- and long-term models.

6.3 Content-Based Media Recommender

Content filtering is a popular technique for classifying items based upon their content, or, as in our case, upon their associated properties. We refer to this process as *Metadata-based filtering*. It relies solely on the information derived from the document's metadata, i.e. the program descriptions. Thus, the outcome of the filtering process relies heavily on the metadata's quality. Inspired by the precision and success of various content filtering approaches, particularly the technique of text classification, we evaluated their applicability to identify programs of interest and generate program recommendations. Based upon a set of experimental data, which includes elements the user likes and dislikes, a user model is derived representing a collection of implicit knowledge (the implicit user profile) about the users' preferences. In contrast to other techniques, such as collaborative filtering, this approach customizes the classification task to each user and, therefore, can adapt individually and automatically to his or her preferences.

Furthermore, the introduction of an explicit profile enables the user to influence the recommendation system's results directly, e.g. by selecting her favorite genres, actors or directors. To provide a simple form of context sensitivity, specific preferences that are contingent upon the situation and may differ from the user's general preferences are added as well. To address potential privacy concerns, the profile is kept on the client side and not shared with broadcasters or EPG data providers.

Within this system component we address the following issues and present feasible solutions for them:

- **The selection of proper classification mechanisms for TV programs.**

- **Gathering, preparation and use of adequate metadata.**

- **Combination of different classification mechanisms for further enhancements.**

- **Dynamic adaptation according to changes in the current situation (available metadata, device, time, etc.)**

The rest of this section is organized as follows. First, section 6.3.1 presents an overview of the most important part of this component, the recommendation engine with its modular structure. Subsequently the parts of our content-based recommendation engine are discussed in the sections that follow. Section 6.3.2 discusses the application of our tokenization approach. In section 6.3.3, we give background information on the overall content filtering process and present different approaches used for single classification steps as well as possible extensions. Then, the explicit user profile is subject to section 6.3.4. Section 6.3.5 covers the combination of recommendations stemming from the explicit and from the implicit user profile. The transfer and adaptation of classification techniques mentioned in section 6.1 to the TV recommendation context is further discussed in section 6.3.6. Before section 6.3.8 concludes the discussion of our content-based recommendation engine, with a short summary and a presentation of possible extensions and of future work, we take a look at several similar projects and discuss how they differ from our proposal in section 6.3.7. A detailed performance evaluation of each classifier and the overall performance of this component is outlined in section 6.5 based upon the collection of our viewing histories as described in section 6.5.1.

6.3.1 Recommendation Engine Overview

One of the most important parts of personalTV is the recommendation engine shown in figure 6.11. A description of the recommendation engine's integration into the overall system architecture, is given in section 6.2.

In our approach, program metadata is linked with the actions of the individual user. These actions are interpreted as a positive or negative indication for the related content. This interpretation is used to learn the users' likes and dislikes. The recommendation engine is trained based upon this data (the viewing history) of the user, i.e., EPG data related to a watched, recorded or skipped program. After the training phase, the engine is able to classify unknown programs. By using their attributes and the program descriptions in the classification step, a value is calculated to represent their relevance for the user. Based upon this value, upcoming programs are assigned to one of two classes - the recommendation and the rejection class. This task is commonly known as a binary classification. Our recommendation engine can use different types of classifiers. Because of the resource-constrained environment of typical set-top boxes and mobile devices, we recommend the use of lightweight classifiers (e.g. our Spam filtering approach) . A typical program description is composed of different elements of information (e.g. start date and time, genre, persons involved, title, synopsis, etc.). Respecting this metadata structure, we propose a setup in which multiple classifiers are trained on specific metadata categories. Thus, each classifier specializes in a specific metadata selection and becomes an "expert" in it.

The classification process, as shown in figure 6.11 on a conceptual level, is divided into the following steps. First, the program metadata elements that are useful for the classification process, such as the synopsis, genre, persons and role names are selected. In the tokenization step (cf. section 6.3.2), these program element descriptions are split into single parts (also known as tokens) identified by delimiters. In most cases, these tokens are single or compound words. Subsequently, the preliminarily trained classifiers calculate a recommendation score, often interpreted as an affiliation probability, of a set of tokens to occur within a recommended program.

The overall program scores are computed as a combination of inputs coming from both, the implicit and explicit user profile, done in the *Score Aggregation* component (further discussed in section 6.3.5). The implicit score is generated by our classifier ensemble Component (see section 6.3.3), whereas the explicit score is gathered by evaluating the explicit *MPEG-7 Profile* (see section 6.3.4) of the user.

6.3.2 Enhanced Tokenizer

The main goal of the tokenization process is to identify tokens from the textual description of the program. In most languages, such as English or German, words can be easily segmented by simply using separator tokens such as spaces, punctuation marks, parentheses and dashes. Thus, simple tokenizer often just make use of these separator elements. Nevertheless, these languages also contain words like contractions ("I'll" or "don't"), possessives ("player's"), foreign phrases ("et. cetera"), numbers or URL's that have to be handled with care. In the context of TV, actors' names and role names like "Brad Pitt" or "Dr. House" need to be handled as compound tokens. Our tokenizer uses a list of words that occur too often

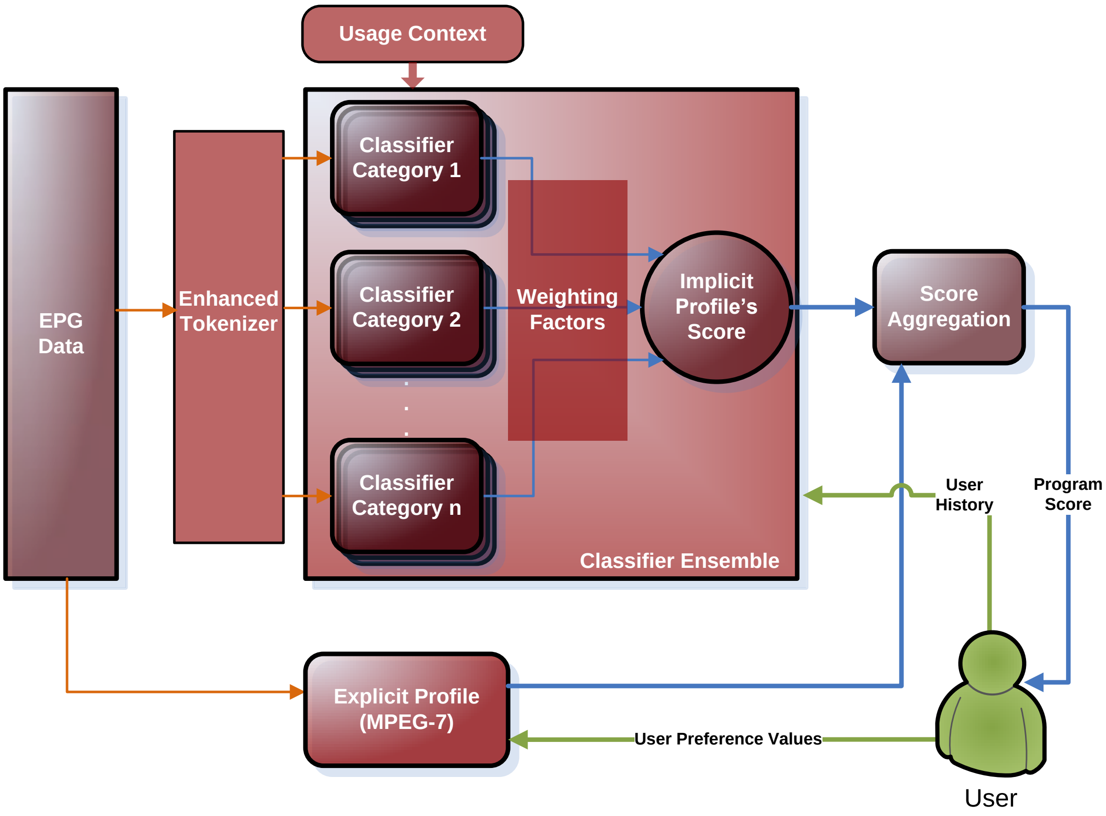


Figure 6.11: Architecture of our content-based media recommender component.

to be of any specificity or that do not possess a significant meaning (the so called stop or noise words). These words are then excluded from the classification process. Short words, without concrete relevance such as pronouns are deleted as well. Inflected and derivationally related forms of words are lemmatized to avoid treatment of varying word forms as different tokens. Due to the syntactical and semantical structure of natural languages, tokens cannot be assumed to occur independently. Thus, token chaining approaches combine multiple tokens, commonly pairs or triples of tokens, which are processed as joint units within the classification step. Especially for groups of words emerging in a specific context, token chaining is expected to yield more promising results than the single token method. For a detailed description of our tokenization framework, please refer to section 6.1.1

Finally, the resulting collection of tokens (bag of words) form the basis for further processing in the *classifier ensemble*.

6.3.3 Classifier Ensemble

In the center of figure 6.11 the classifier ensemble with its multiple classifier setup (further discussed in paragraph "3. Classifying" in this section) and the system components responsible for each step of the process are shown. The combination of these classifiers forms a group of experts that we refer to as a *classifier ensemble*. In most cases, these ensembles are able to generate more precise recommendations than a single classifier trained on all metadata elements would do. The value of the implicit profile is obtained by combining each classifier's score balanced by a weight, which represents the different level of priority of the metadata elements. The whole classification process consists of the following three steps:

1. linking of programs with user actions
2. learning the relevant program descriptions with respect to related user actions
3. classifying programs based upon learned program descriptions

1. Linking:

Based upon the viewer's usage history, the single actions and their intervals are aggregated and linked with the related programs. User actions (e.g. switching channels) within a short interval (e.g. zapping) are ignored or included in a very "lightweight" manner, which counterbalances their effect. To take into account the possibility of users switching the channel to avoid commercial breaks, programs watched for over 80% of the program's length are regarded as completely watched programs. This also applies to programs that the user started to watch right at the beginning of his or her TV session, assuming he or she has missed the beginning of the program. Reducing the weighting of partially watched programs is a further option to refine the interpretation of user actions. Depending on the classifier settings, the programs that have not been watched during the viewer's TV-session (concurrent programs) are included as "lightweight" negative samples.

2. Learning:

The programs with their associated user actions are processed by the classifiers. In the context of content-based classifiers, the program descriptions are categorized as watched, not watched, recorded, accepted recommendations and rejected recommendations. These actions indicate different grades of likings which are represented by adequate multipliers (cf. default weights in table 6.3) within the learning process. In our first evaluations, we discovered that these factors heavily depend on each individual user. Thus, these factors are modeled as variables and initialized with default values. Similar to other classification tasks, in our approach the balance between learned rejections and learned recommendations is of high importance for unbiased recommendations. Thus, the default weights are modeled with the goal of keeping both sides balanced and are derived from pre-experiments. Table 6.3 shows how user actions are interpreted by our system. Depending on the user's choice, recent user actions can be included immediately or on a regular basis such as daily or weekly. Thus, the profile dynamically adapts to the user's evolving viewing habits.

3. Classifying:

Based upon the data gathered at the learning step, single programs can be evaluated. Different types of classifiers can be used for this. Within our system several classification approaches have been implemented and tested for applicability in recommendation systems (cf. section 6.3.6). As far as the classifiers' input is concerned, we propose the following variants:

User action	**Description**	**Default weight**
watched program or recorded program	a program that has been watched for a minimum of 10 % of its duration	z * x
skipped program	a program that has been watched for less than of 10 % of its duration	1 * y
accepted recommendation	a program with a score indicating interest that has been accepted (thumb up in program ranking table)	10 x
rejected recommendation	a program with a score indicating interest that has been refused (thumb down in program ranking table)	20 y
query key words	phrases used in the full text search box of the remote control	10 x
zapping periods	phases when programs are watched less than a specified interval	1 * y

Table 6.3: Interpretation of user actions during a TV session (z = all programs that have not been watched while watching another program, x = learn as recommendation, y = learn as rejection).

- [*1 classifier 1 metadata element*] a single classifier relies on data from one metadata element (in the training and the classification step).
- [*1 classifier n metadata elements*] a single classifier is used for an element combination. Commonly related elements such as persons participating in a program like actors, director, moderator, guests are merged.
- [*1 classifier + prefix*] a single classifier is used on prefix tagged data. This means that every part of the input is tagged with a prefix indicating the related metadata element.
- [*m classifier n metadata elements*] multiple classifiers are used based upon the data of one or even multiple metadata elements. This approach is able to cope with the different demands of users, especially to provide adequate recommendations under special conditions e.g. varying importance of elements or varying preferences in situations such as weekday vs. weekend or morning vs. evening. This enables the recommendation system to take the findings of [Ber08] into account, which showed that the time context of recommendations is of major importance. In this setup, each classifier is designed to address single requirements and is specialized in specific metadata elements. The results are combined, using individual weighting factors for each classifier. Weights can be set uniformly or by experimentally determined presets. In section 6.3.6, a dynamic weighting approach is outlined. Due to the modular structure of our recommendation engine, different classification approaches may also be used simultaneously within this setup.

Commonly the score of the implicit profile $I(p_i)$ for a program p_i is calculated as follows:

$$I(p_i) = \sum_{j=1}^{N} c_j(p_i) * w_j \tag{6.29}$$

with $c_j(p_i)$ being the score value of classifier j in the range of [0,1] and w_j being the weighting factor of classifier j under the condition

$$\sum_{j=1}^{N} w_j = 1 \tag{6.30}$$

6.3.4 Explicit Profile

The explicit profile only contains values entered by the user. Based upon MPEG-7, it enables the user to directly configure his or her viewing preferences by selecting favorite genres, actors and so on (see figure 6.8 - profile definition dialog). The available options for preference selection are gathered from the program metadata and its schema. Frequently used tags can also be used in this step. The profile is built upon the selected elements or tags and a preference value y with $y \in [-100{,}100]$ where -100 denotes no interest and 100, great interest. For upcoming programs a direct string matching operation against components of the program description is conducted. In case of a match, the specified preference value is attributed to the program. The overall score of the explicit profile $E(p_i)$ of a specific program p_i is calculated as follows:

$$E(p_i) = \frac{1}{N} \sum_{j=1}^{N} y_j \tag{6.31}$$

with N being the number of matching preference elements and y_j being preference value of the matching element.

6.3.5 Score Aggregation

The overall score $O(p_i)$ of a program is calculated by merging scores from the implicit and explicit profile as follows:

$$O(p_i) = I(p_i) * w_I + E(p_i) * w_E \tag{6.32}$$

where w_I denotes the weight of the implicit profile respectively w_E the explicit weight with $w_I + w_E = 1$. In the case of multiple classifiers, the implicit profile's result is obtained by combining each classifier's weighted score. The weighting factors of the profiles specify their impact on the program score. Preferences stemming from the explicit profile are used as guidelines within the cold start phase to overcome the problem of providing inappropriate recommendations due to insufficient data in the implicit profile. Hence, during the cold start phase, the explicit profile's weighting is amplified to improve the recommendation quality. As more usage history data becomes available, the explicit profile's impact on the program rating is linearly reduced by default. By amplifying the explicit profile's weight, the user is still able to adjust the system's behavior towards a recommendation that relies more on his or her explicit preferences. A dynamic weight adjustment as proposed in paragraph ”Dynamic Weight Determination for Classifier Ensembles” in section 6.3.6 could also be used for balancing explicit and implicit profiles. Additionally, the user can control the two profiles' impact on the overall result by adjusting the related weighting factors in the remote control's profile preference settings.

If the overall score of an upcoming program exceeds a certain threshold value for unconditional recommendations, the program gets recommended. In the case that the user ignores or misses such a broadcast of special interest, it is either auto-recorded or just ignored depending on the settings. Another enhancement for a more precise distinction between different kinds of recommendations and rejections can be made by taking the confidence of classifier scores into account (cf. paragraph ”Confidence Values” of section 6.3.6).

6.3.6 Application and Adaptation of Classification Approaches

Within this system, we transferred and adapted the following dynamic approaches to fit the context of TV recommendation systems:

Spam Filtering Approaches

The main concepts of spam filtering and different statistical filtering approaches have been introduced in section 6.1.2. Due to the ”lightweight” character of statistical filtering, these approaches are very suitable to resource-constrained environments and therefore to TV set-top boxes. Even if this approach is quite simple, its efficiency in classifying text is roughly comparable to that of other, more resource consuming and complex methods such as Support Vector Machines (SVM) (cf. Table 13.9 [Man08]). According to [Man08] and the results of [Ira04] a classifier with high bias, such as Naive Bayes, as used in spam classification, is a good choice in situations where a supervised classifier is trained using fairly little data. We face this very situation in the realm of TV.

Supported by these arguments, the application of spam filtering based classifiers seem to be a good choice for the TV recommendation generation. These approaches are used in the following manner: Based upon the content of program metadata, a procedure similar to classifying emails into spam and ham has been derived. The idea is to classify upcoming programs (incoming mail) according to the user's preferences, as identified by his or her past actions. In comparison to the two actions ”accept” or ”reject” as seen in the context of email classifiers, the domain of TV includes multiple actions: rejecting a recommendation, not watching a program, watching a program, recording a program and accepting a recommendation. These user actions are categorized into two classes:

- acceptance / recommendation (positive user actions): watched programs / recorded programs / accepted recommendations
- rejection (negative user actions): not watched programs / rejected programs

The main issue stems from the simultaneous broadcast of programs: with email, each message is processed by the user, whereas most programs are not watched and therefore not judged by the user. This results in an huge amount of unclassified programs and as a result introduces the problem of obtaining reliable rejections. This problem can be at the very least limited by slightly devaluing unwatched programs, where the broadcasting time overlaps with a watched or recorded program, and considerably increasing the value of watched programs and accepted recommendations.

Compared to spam filtering, in our approach the spam class corresponds to the recommendation class and the ham class corresponds to the rejection class. This may seem counter-intuitive. The reversion is motivated by the following properties of spam filtering and the program recommendation problems:

- **intuitivity:** To enable easy human interpretation of results, they are presented in a range from 0.0 to 1.0 with 1.0 indicating a program of high interest and 0.0 the opposite case.

- **false-positives:** In the context of spam, avoiding false positives means avoiding misclassifying ham emails as spam, and in the TV context, avoiding classifying programs which do not match the users interests, as recommendations.

- **false-negatives:** It is intended to eliminate false negatives (spam in the in-box / interesting programs not on the recommendation list) by increasing the system's discriminatory power.

- **clear distinction:** An intelligent classifier selection should be able to adequately categorize programs into recommended / not recommended even if they have very similar descriptions. Year of creation, country, actors or role names are considered to be promising elements that can be used to differentiate more clearly in such cases (cf. results shown in section 6.5.4).

Support Vector Machines (SVM)

SVMs, as introduced in Section 6.1.3, can be used to complete a wide range of tasks. For each specific area of application, the task must be presented in a way that is conducive to the use of SVMs. In our case, for a typical text classification task with the special characteristics of the TV domain, the data has to be mapped to fit this mechanism. In the following section, we will present our approach of using a SVM as a classifier in our system. The application of a SVM to text classification and categorization has already been addressed in several publications, such as [Joa98, Joa99, Seb02, Gab04]. Within our approach, the findings of these publications have been considered as basic guidelines. For a well written guide how SVMs can be applied to different problems, the interested reader may refer to [Hsu03]. The process of problem formulation can be roughly organized into the following steps:

1. Data Selection and Pre-Processing: In order to apply SVMs to a problem, the dataset has to be represented in a fixed numerical vector form. Each sample drawn from the dataset is mapped in this common vector form. In our case, program metadata has to be mapped to a vector. Commonly, categorical elements with a small and fixed number of possible values, such as genre or category, are easy to handle and to represent. In contrast to these easy to deal with elements, free text elements,

such as synopsis or elements with an unlimited number of values, such as actors or persons, are rather problematic. Due to the large amount of available metadata, a well developed selection of metadata elements and element values represented in the vector should be made. For these free text elements, a pre-processing step is conducted to reduce the number of available variations for each item. Among others a stop-word reduction and stemming is made in our approach. Only taking into consideration tokens with a specific grammatical category, as has been done in [Xu06] would certainly be an easy way for further reductions. At the end of this step, a selection of reduced metadata is produced, which can then be used in the construction of an input vector.

Based upon the selection of the pieces of metadata, a further decision related to the issue of ”document collection” must be made. In this part of our approach, we make a distinction between two types of collection:

a) Common document collection: All available program metadata is used in vector construction. Thus, this vector is universally applicable for all users and is able to represent every program in the collection. Although this vector features several advantages, a major drawback is the amount of time necessary for processing millions of program descriptions. Furthermore, it is very hard to find an acceptable trade-off between the size of the vector and the quality of program representations enabled by this vector. This approach leads to a sparse vector representation for any specific program.

b) User specific document collection: The vector construction is based upon a small subset of the program data collection, the viewing history of a specific user. Typically all concurrent programs are also considered in this construction step. Because of the very limited nature of such a dataset, the vector can be built efficiently. The size of the vector depends solely on the number of programs in the user's viewed history and the number of concurrent programs. Nevertheless, this approach also has a drawback. It is only able to represent programs with wording similar to that of programs already seen by the user and the concurrent ones. Thus, most programs could not be appropriately represented. Because of this, it would not be possible to represent a fundamental change of user's interest since new programs watched would not fit into the constructed vector.

2. Transformation and feature extraction: The main purpose of this step is the transformation of EPG data into a concrete vector representation. For the representation of text, we decided to follow the suggestions of [Leo02]. Due to the fact that the input vector needs to be provided in a numerical form we need to map our textual elements to numerical values. This transformation process is commonly composed of three steps.

   - First a frequency transformation is used to indicate the importance of a term t_i in a document d_j. Term frequency is often defined by bijective mapping $tf(t_i,d_j)$ measuring the number of occurrences of term t_i in document d_j. For most text categories the frequency of different terms greatly varies, following, to a certain extent, the well known Zipf-Mandelbrot law. Thus, several terms such as articles occur very often, whereas the main portion of tokens occur very seldom. In order to efficiently transform information pertaining to frequency into a representation usable by a SVM, several methods have been proposed. The simplest forms are the boolean representation and the direct use of the frequencies, also called raw term frequencies or term frequencies (tf). The transformation f_t is defined as follows:

$$f_t(t_i,d_j) = f(t_i,d_j) \tag{6.33}$$

Whereas the boolean representation for $f_t(t_i,d_j)$ is modeled as follows:

$$f_t(t_i,d_j) = \begin{cases} 1 \ if \ f(t_i,d_j) > 0 \\ 0 \ else \end{cases} \tag{6.34}$$

A widely used form in the realm of linguistics is the logarithmic transformation:

$$f_t(t_i,d_j) = log(1 + f(t_i,d_j)) \tag{6.35}$$

In the mapping of frequencies to the unit interval, the inverse term frequency transformation is often used:

$$f_t(t_i,d_j) = 1 - \frac{\gamma}{f(t_i,d_j) + \gamma} \tag{6.36}$$

The parameter γ is often set at 1 to start, thus ensuring that zero frequencies are not mapped as a well defined value.

- In the second step, a weight, based on the importance of different terms, is introduced. Because not all terms can be considered to be equally important, one way to distinguish between more and less discriminating terms in the documents must be found. It is clear that terms occurring in nearly every document in a collection are less important and less discriminative than a term that occur only in one, or only a few specific documents. The main idea is to reduce the weight of terms with a high collection frequency. Commonly two weighting measures, inverse document frequency (idf) and the redundancy weighting, are used. With SVMs, the idf is the most popular method for term weighting. For a term t_i and the total number of documents in a collection N it is defined as follows:

  $$idf_i = log\frac{N}{df_i} \tag{6.37}$$

  df_i stands for the count of documents where term t_i occurs. idf assigns high scores to terms occurring in a small number of documents, whereas the score of very frequently occurring terms is low. The redundancy weight r_i for a term t_i is defined as follows:

  $$r_i = logN + \sum_{j=1}^{N} \frac{f(t_i,d_j)}{f(t_i)} log\frac{f(t_i,d_j)}{f(t_i)} \tag{6.38}$$

  $f(t_i)$ measures the frequency of term t_i in the entire collection of documents.

It measures the derivation of term t_i's distribution in relation to the uniform distribution. Compared to the idf, the redundancy weighting also accounts for the frequency of occurrence of a term in a specific document ($f(t_i,d_j)$).

To produce a composite weight of term frequency and importance weight, both measures are combined by multiplying the values. In general, each frequency transformation can be combined with each importance weight. Among other combinations, one of the most popular is tf-idf. According to [Man08], this is defined as the combination of raw frequency and the inverse document frequency:

$$tf - idf_i = f(t_i,d_j)idf_i \tag{6.39}$$

tf-idf assigns a high value to terms occurring very often in a specific document and rarely in other documents. These terms have a very high discriminatory power between documents. Inversely, terms with a high collection frequency get a very low value.

- The transformation process is concluded by a normalization step. The normalization accounts for the different length of documents, as defined by the number of terms in each document. This makes documents with different token counts comparable. Typically, the type-frequency is divided by the token count. Common approaches are L1- and L2-normalization (see [Leo02] for details).

3. Training of the user model: After transforming documents into the vector representation of the SVM, the next issue concerns the proper selection of training data for both, the negative and the positive classes. One Class SVMs have not been considered in our system, as they commonly feature a very low level of classification precision (cf. [Li03]). The selection of a specific SVM type and a kernel is here considered to be an issue of empirical evaluation and, therefore, covered in section 6.5. Positive samples are selected in the same way as it has been done with our other classification methods, based upon the viewing history of the user. Compared to the spam classification based approach, the task of dealing with SVMs is much more delicate, because training samples can not be weighted differently and in a binary classification task both class have to be represented in the training. Because of this, even more attention must be paid to the proper selection of negative samples. We investigated two different methods for this type of selection. The first method suggests the selection of concurrent programs for each program of the viewing history as negative examples. Concurrent programs are all programs with overlapping broadcasting times in relation to a specific program in the history. Due to the huge count of available channels in the dataset also the channels considered in the selection of concurrent programs have been limited to those included in the user's viewing history. The second method offers a more systematic selection of negative samples based upon the mechanism of Rocchio SVM (RocSVM) [Li03]. Compared to other approaches for typical positive unlabeled (PU) learning tasks such as Spy-SVM [Liu02], PEBL [Yu02] or Biased-SVM [Liu03], RocSVM is one of the best methods and widely used (cf. [Li03, Qiu09]). A set of positive samples P and a set of unlabeled documents U is given. The main goal is to split U into a set of "reliable negatives" (RN) and a set of unlabeled samples. For this purpose a

positive (c^+) and a negative (c^-) prototype vector is created in the following way:

$$\vec{c}^{+} = \alpha \frac{1}{|P|} \sum_{\vec{d} \in P} \frac{\vec{d}}{||\vec{d}||} - \beta \frac{1}{|U|} \sum_{\vec{d} \in U} \frac{\vec{d}}{||\vec{d}||} \tag{6.40}$$

$$\vec{c}^{-} = \alpha \frac{1}{|U|} \sum_{\vec{d} \in U} \frac{\vec{d}}{||\vec{d}||} - \beta \frac{1}{|P|} \sum_{\vec{d} \in P} \frac{\vec{d}}{||\vec{d}||} \tag{6.41}$$

According to the findings of [Buc94], α is set to 16 and β to 4, adjusting the impact of negative and positive labeled documents. Based upon these vectors the RN set is constructed by the union of documents d, satisfying the following condition:

$$sim(\vec{c}^{+}, \vec{d}) \leq sim(\vec{c}^{-}, \vec{d}) \tag{6.42}$$

The similarity is measured by a cosine distance measure. For the document vector representation, a common tf-idf approach is applied. To further refine RN, the following process is used: Based upon RN and P, different SVMs are trained in several iterations. In each iteration a new SVM is trained on the selected document sets. Documents in a set Q, initialized as $Q = U - RN$, are classified and a set W of further negative samples is constructed. Accordingly, Q is set to $Q = Q - W$ and RN to $RN \cup W$. The process stops when no further documents are classified into the negative class. The choice of which SVM and of which training set to use is dependent on the percentage of erroneously classified documents of P. If the level of error is lower than 5% the first is chosen, in all other cases, the last SVM, is used as the Rocchio SVM. A further enhancement of the Rocchio step is possible by introducing an additional clustering step. For a detailed discussion see [Li03].

4. Determination of adequate SVM parameters: Depending on the selection of a specific SVM type and kernel, different parameters have to be set properly in order to achieve good results. For example, the parameter C and γ must be determined for a RBF kernel, and γ and r for the sigmoid kernel. The accuracy of a parameter setting is determined by a n-fold cross-validation step as described in section 6.5.2. Cross-validation is used, because the repeated and intersecting splitting of the dataset into a training and a classification part guarantees a higher accuracy of the classification results, and better reflects the achievable generalization ability. Through the use of multiple iterations with the permutation of trainings and classification parts, overfitting effects can be avoided. The grid search approach is a very simple and a straightforward parameter search method. It is a widely used and very reliable method for determining SVM parameters. In the parameter space, a uniform grid is defined where each point represents a specific parameter setting. Parameter estimation in our system has been completed using a coarse grid in the first step and then refining the grid in the regions of interest. A Grid has been used, mainly because it can be concurrently processed. Experimental results are shown in section 6.5.3.

5. Classification of upcoming programs: Beside the class labeling task, the results of the classification process can also be interpreted as a affiliation probability towards

a specific class. In [Pla99] Platt et al. propose the following method for measuring the posterior probability for a decision value:

$$P(y = 1|f) = \frac{1}{1 + exp(Af + B)} \tag{6.43}$$

with A and B set by a prior maximum likelihood estimation from a set of training samples. In our approach this method has been used to provide affiliation probabilities for TV programs towards a specific class.

6. Updating of the model: Because of the nature of our classification problem, model updates are needed frequently as the profile (viewing history) of a user evolves. A SVM can commonly learn to incorporate new samples in iterative learning steps and therefore no new mechanism is needed to update the model. However, in the realm of TV recommenders, this is not the case. With the introduction of new upcoming programs new terms such as new actors, directors or even new words are introduced in the program descriptions. Thus, the vector representation must also be updated from time to time to allow for a proper representation of new programs. SVMs are commonly incapable of handling frequent updates of the vector representation, so after changing the vector also a new training of the entire model is needed. To cope with this problem, one possible solution would be the development of a special kernel or a preprocessing step producing for all text documents a common vector representation. In our approach we decided to update the vector and conduct a new training step on a certain window of the viewing history. We suggest that in the majority of cases, not all programs from the viewing history are relevant for a proper classification of upcoming programs. Furthermore, the inappropriateness of the vector for representing new programs can be easily measured based upon the count of newly extracted features that can not be mapped. This measure can be used as a trigger to initiate the updating process. Depending on the selection of the document collection, updates have to be done more or less frequently. Commonly, the construction of a individualized vector for a specific user leads to faster degeneration of the representation, whereas the common vector form can be considered to be more stable. This issue has been also addressed in our evaluation presented in section 6.5.3.

For measuring the confidence of classification results, SVMs provide a kind of built-in mechanism - the distance of the current sample to the decision hyperplane. This distance can generally be seen as an indicator for the reliability of the classification. If the sample is close to the decision boarder the probability for faulty classification is higher than in such cases where the distance is very large. Thus, this mechanism can be used to enhance the reliability of program recommendations.

LSI Based Classifier

Latent Semantic Indexing (LSI) is a very popular technique used in a wide variety of fields. Especially in the realm of information retrieval, it is frequently used for tasks such as reducing the size of search indexes, and for making the systems faster and easier to handle. In order to apply LSI in the field of television, three main issues have to be tackled:

1. **Construction of the model**: To begin the LSI, the term-document matrix A must be built. The model is represented by the different matrices $\mathbb{U}$, Σ and V (cf. section 6.1.4). The matrix A is commonly dependent on the number of documents and the number of different terms each document contains. In our case, we are faced with 164 different channels, and more than 5000 programs broadcast each day. Confronted with this amount of programs, runtime of the SVD becomes a big issue especially when the model is constructed using all available programs. Preliminary runtime tests on a typical desktop PC showed that the process of decompounding a 5000×5000-Matrix into its submatrices takes up to seven hours. Please note that the memory consumption of such matrix must be also considered. Due to resource constraints on set-top boxes and time constraints on the recommendation generation, feasible restrictions for the matrix size must be found. A way to cope with this issues is a restriction based upon the time frame, the metadata structure of the programs and the number of channels considered within the constructed model. Based upon these options, the following distinctions can be made:
   - Reduction of the number of programs: An obvious way to achieve this kind of reduction is to limit the time frame of programs considered in the constructed model e.g. to one or two months. Please note that short time frames commonly tend to intensify the need to frequently update the model. Moreover in most cable and especially terrestrial networks, only a limited subset of our 164 channels is available. Channels considered can also be limited to the most popular ones. For instance, taking only the ten most popular channels of our dataset (cf. table 6.4) into account would reduce the number of programs to approximately 6 % of the total available programs.
   - Reduction of the number of terms: Based upon different metadata elements of programs, different models can be constructed for specific program groups. For instance, a program genre specific model is one option to reduce the amount of data. Specifically, it would limit the number of different terms used in the program descriptions. Under such limitations, the reduction mainly depends on the specific genre of the programs. Measured on our dataset (cf. section 6.5.1) the reduction potential of a genre specific model varied between up to 90 % in the news genre to 25 % in the movie genre compared to the term count of a collection of programs without further grouping. Further reductions are expected because of the additional use of our enhanced tokenization framework (cf. section 6.1.1). Only taking tokens of a specific grammatical category, such as nouns, into account is another feasible way to reduce the number of terms.
   - Reduction of both: In our case, the LSI model is used to identify similarities between programs of the user's history and other programs, in order to identify programs potentially interesting for him. A kind of user specific model constructed solely based on the user's history would also enable a distance measure in a very similar way. Moreover, with this approach the dimensionality of the term-document matrix only depends on the size of the user's profile, which is typically rather small.
2. **Updating of the model**: Because every upcoming program has to be considered as

new item introduced into our text corpus the model needs to be frequently updated. Especially when new programs are added to the user profile, the model would not be able to keep track of the change in user habits without adequate updating. In the following paragraphs, several methods for model updating will be presented. These methods generally differ in the precision of the updated model and in the complexity of calculation.

- Recomputing: The easiest way is the full recomputation of the model because existing methods for building the model can be fully reused. This method also leads to the most accurate model and its model can be considered as the baseline compared to other updating methods. A serious drawback of this method is that it is very time consuming and complex to calculate.
- Folding-In: This approach, presented in [Dee90], facilitates the introduction of new documents into the model quickly in a similar way to the query mapping step described in section 6.1.4. First the new document $\vec{d}_{new}$ is transformed into the k-dimensional vector space of the model by the following equation:

  $$\vec{d}_{newk} = \Sigma_k^{-1} \mathbb{U}_k^T \vec{d}_{new} \tag{6.44}$$

  Accordingly, the projected document $\vec{d}_{newk}$ is added to the bottom of the reduced document representation matrix V_k. For each new document in a collection, an incremental update step is conducted. Although this method performs rather well, the document representation degenerates with each document added, and becomes less and less accurate. This is mainly due to the inability of the updated model to reveal latent dependencies between the new documents. Moreover, even new terms, not present in the current index, can't be mapped to the model and are therefore ignored. Although [Dum95] reports a satisfiable precision of folding-in in practice, the precision still remains a recurring problem in the realm of TV programs because of heavily varying description texts and therefore a very fast degeneration of the document representation. Commonly, the matrices V_k and Σ_k are not changed at all within the folding-in process. Pre-experiments on a small test set showed that folding-in is about ten times faster than recomputing.
- Updating: This method, introduced in [Zha99], updates the model in a more accurate, although more complicated way. Compared to folding-in, it modifies all matrices $\mathbb{U}_k$, Σ_k and V_k according to the new document and therefore covers the document, term and the term weight update problem. The document update process is conducted after adding new documents D to the term document matrix A_k given $B \equiv [A_k, D]$ and $A_k = P_k \Sigma_k Q_k^T$. Then, after a QR decomposition, the updated and best rank-k form B_k is given by:

  $$B_k \equiv ([P_k, \hat{P}_k]\mathbb{U}_k)\hat{\Sigma}_k \left(\begin{bmatrix} Q_k & 0 \\ 0 & I_p \end{bmatrix} V_k \right)^T \tag{6.45}$$

  when $\hat{P}_k$ is orthonormal. Term update is done very similar to the document update. New terms T are added to A_k given that:

$$C \equiv \begin{bmatrix} A_k \\ T \end{bmatrix} \tag{6.46}$$

Then the approximated form is defined as follows:

$$C_k \equiv \left(\begin{bmatrix} P_k & 0 \\ 0 & I_q \end{bmatrix} \mathbb{U}_k \right)^T \hat{\Sigma}_k([Q_k,\hat{Q}_k]V_k)^T \tag{6.47}$$

Term weight update is done based on the term weight correction matrices in $W = A_k + Y_j Z_j^T$ where Y_j specifies a selection matrix containing 1 for terms where a term weight correction is necessary and Z_j^T containing the weight difference between the old and the new terms.

$$W_k \equiv ([P_k, \hat{P}_k]\mathbb{U}_k)\hat{\Sigma}_k([Qk, \hat{Q}_k]V_k)^T \tag{6.48}$$

As a result of the updating process covering all matrices of the model, it produces the same results as the recalculation, with the exception of some rounding errors. Updating features a significantly lower computational expense whereas the expense of folding-in is still much lower than those of updating.

- Folding-Up: Folding-up [Ber95] is a hybrid approach combining the method of folding-in with updating. First the update procedure uses folding-in of new documents until a certain threshold defined by a pre-selected percentage of the document-term matrix. The threshold should be individually selected with respect to the area of application, taking into account the different impacts of more or less varying documents on the model's precision. After the threshold is reached, all documents previously folded-in are discarded from V_k and an update step is conducted. Based on the updated matrices $\mathbb{U}_k$, Σ_k and V_k the process starts again at the first step. Folding-up features the same complexity as updating in the updating steps, but within the boundary of the threshold as folding-in. This lower computational expense comes at the price of buffering all document vectors between the update steps and the need to empirically select the threshold. Pre-experiments on a small test set showed that folding-up is (dependent on the threshold) about two times faster than recomputing.

Note that all mentioned methods are also able to introduce new terms into the model in a very similar way e.g. in equation (6.44) by simply substituting document and term. For a detailed evaluation and comparison of all mentioned update mechanisms the interested reader is referred to [Zha99, Tou07].

3. **Scoring and ranking of programs**: LSI based text classification has been already used in various ways for different purposes. Most approaches use LSI for clustering purposes. Several LSI document classification approaches have been proposed. These can be distinguished based on their method - either global LSI or local LSI. Global LSI computes the LSI feature space based upon the entire dataset. As a fully unsupervised method, it does not take into account the data's class labels. Using this

model, the classification task is commonly conducted by simply counting the class labels of k documents with the highest cosine similarity and assigning the majority class label to the new document. In contrast to the way global LSI approaches handle the dataset, in local LSI approaches the dataset is further separated based upon features of the dataset such as genres or topic. Thus, for each data subset a separate LSI step is conducted and an individual LSI feature space is built. This method generally has a superior precision compared with global LSI. In [Din08] an advanced approach is presented and compared to both, the global and the local method. In this approach, text is classified into several categorize based upon an index containing the most representative text examples of each category. Representative examples are selected by a preliminary classification step. Compared to local and global LSI, the index is much smaller because of the further reduced data samples considered in the LSI step. Although these approaches feature good experimental results, they are hardly applicable in the realm of TV recommendation. In general, the user's preferences are not restricted to a certain category or genre of programs. Thus, most profiles are scattered over the whole semantic space after indexing. Furthermore, the computational expense in terms of time and resource consumption for the LSI step conducted on the whole program collection would be very high. Upon these preconditions we decided to develop our own classification approach solely founded upon the viewing history of the users. LSI is performed solely on the user's viewing history building a semantic space for programs considered to be of interest for the user. Although within the low-rank approximation the term-document matrix contains only programs that have been already watched and their respective descriptive terms, we assume that the main concepts of interest are included. For the purpose of ranking upcoming programs the k-nearest neighbors for each program are determined based upon the cosine similarity measure. The programs are then ranked based on their count on the different levels of similarity. A high number of neighbors can generally be considered as an indication of similarity to the user's profile. Besides the purpose of ranking, classification scores are often needed for subsequent processing steps. A naive approach would be to use the average similarity measure of the samples:

$$score(p) = \frac{1}{K}\sum_{i=1}^{K} sim(p,p_i) \tag{6.49}$$

when p is the program of interest and p_i is the i-th neighbor of the k nearest neighbors. In this approach all neighbors are treated the same. This may cause problems in cases where, for instance, a program p_1 (e.g. ”Terminator 2”) has only one neighbor with a high similarity of 0.95 (e.g. ”Terminator 1”) and three neighbors with 0.3 gets a lower average than an other program p_2 with four neighbors with a score of 0.5 each. In order to counterbalance the weight of samples with very few "good" neighbors the application of a confidence function should be considered. Another solution to this issue can be found by using a modified approach to the average weight used as shown in equation (6.50).

$$score(p) = \frac{1}{\sum\limits_{i=1}^{K} sim(p,p_i)} \sum_{i=1}^{K} sim(p,p_i)^2 \tag{6.50}$$

Due to the squared similarities in the summation, high similarities are amplified leading to a slightly higher score for the program p_1 compared to p_2 in the example.

Dynamic Weight Determination for Classifier Ensembles

As mentioned before, the recommendation engine must group a series of classifiers into a classifier ensemble to provide an estimation for TV programs of interest. The various scores are weighted and linearly combined. Manually configured weighting factors would guarantee an optimal combination of classifiers for an individual user. Nevertheless, it is not convenient for large classifier setups to do this manually. The problem can be formulated as shown in equation (6.51).

$$\begin{aligned} \min \sum |e_j - u_j| \;\; with \;\; e_j = \sum_i w_i c_{ij} \\ 0 \leq w_i \leq 1 \; and \; \sum_i w_i = 1 \\ w_i \ldots Weightingfactor \; of \; the \; classifier \; i \\ c_{ij} \ldots Output \; score \; of \; the \; classifier \; i \;\; and \;\; 0 \leq c_i \leq 1 \\ u_j \ldots User \; feedback \; for \; the \; sample \; u_j for \; program \; j \;\; with \;\; u_j \in \{0,1\} \\ e_j \ldots Linear \; combination \; of \; classifier \; scores \; for \; program \; j \;\; with \;\; 0 \leq e_j \leq 1 \end{aligned} \tag{6.51}$$

There are several possible solutions for this issue. A fast and simple way is the application of a step function $T(c)$, which is used to assign bonus or malus values to each classifier based upon their classification scores c. The calculation of the classifier weights is done as shown in equation (6.52).

$$w_{i,j+1} = \frac{\sum\limits_{j} T(c_{ij})}{\sum\limits_{ij} T(c_{ij})} \tag{6.52}$$

where $T(c_{ij})$ denotes the bonus/malus value for a classification value of classifier i for program j. $w_{i,j+1}$ is set as the new weighting factor for classifier i. As a step function

$T(c)$ assigns a value between -5 and 5 to each classifier.

$$T(c) = \begin{cases} 5 \ if \ c > 0.9 \\ 4 \ if \ 0.8 < c \leq 0.9 \\ 3 \ if \ 0.7 < c \leq 0.8 \\ 2 \ if \ 0.6 < c \leq 0.7 \\ 1 \ if \ 0.5 < c \leq 0.6 \\ -1 \ if \ 0.4 < c \leq 0.5 \\ -2 \ if \ 0.3 < c \leq 0.4 \\ -3 \ if \ 0.2 < c \leq 0.3 \\ -4 \ if \ 0.1 < c \leq 0.2 \\ -5 \ if \ c < 0.1 \end{cases} \tag{6.53}$$

A better, more sophisticated way to differentiate between the bonus/malus value of very high (close to 1) and very low (close to 0) classification scores is offered by the application of the *logit* function. A plot of the function is shown in figure 6.12. Where $c_{ij} \in]0,1[$ has to be assured.

$$logit(c_{ij}) = log \frac{c_{ij}}{1 - c_{ij}} \tag{6.54}$$

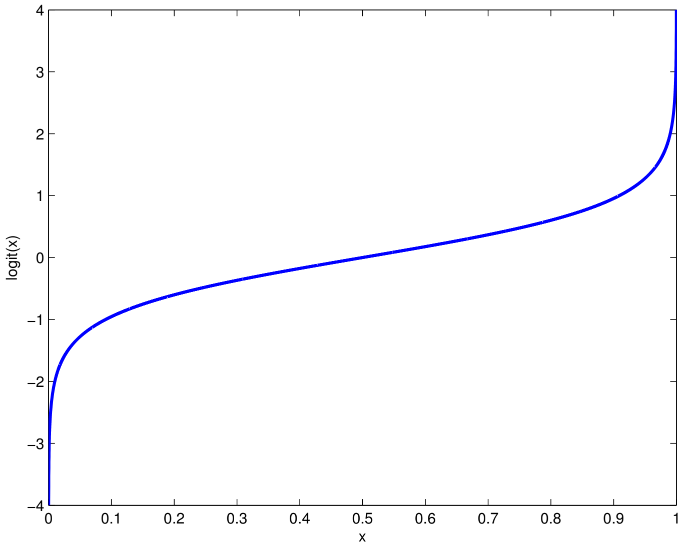


Figure 6.12: Plot of the logit fuction.

Another way is to compute the weighting factors by applying linear optimization. To determine the weights, a section of the viewing history - the optimization window - is selected. The minimum size of this window is defined by the number of classifiers in the ensemble. By changing the window size, the dynamic of the weights can be modified. For example, enlarging the window usually leads to a steadier weighting factor. The cold start phase is fixed by the window size. If no selection of important elements is conducted, all metadata elements with uniformly weighted classifiers will be taken into account on the startup. When a sufficient amount of watched or recorded programs is available in the history, the weighting factors can be recalculated. At this stage, only positive actions are taken into account. Such actions are always valued with 1.0. Thus, the absolute value in equation (6.51) could even be skipped. The main goal here is to minimize the total difference between predicted values and the real values indicated by the users' actions. When the next action (the watching or recording of a program) is added to the history, the optimization window is shifted further and the weights are optimized again. As the optimization progresses, classifiers, using a metadata element which has greater influence on the users interests are amplified. For example, if someone is fond of a certain actor, the classifier related to this element will be of higher importance. Based upon this idea, classifier sets are used to partition the element space to provide more accurate and context specific recommendations. Suitable metadata elements for partitioning are the current weekday, as well as genres in a discrete attribute space and daytime divided into several intervals (e.g. morning, noon, afternoon, evening) in a continuous attribute space.

Another possible approach is the use of a typical meta-classifier as described in section 4.4. The meta-classifier is directly trained using the classification results of the individual classifiers, commonly referred to as base classifier, of the classifier ensemble. The meta-classifier is used to predict the correctness of each individual base classifier. Thus no explicit weighting factors need to be determined. As meta-classifier most classification approaches such as Support Vector Machines or decision trees can be used. We decided to use a simple biased feed-forward neuronal network in our approach, as it offers a flexible, resource conserving and easy way to evaluate the meta-classification approaches in our system.

Confidence Values

The reliability of a sample's or word's affiliation towards a category considerably impacts the quality of classification results. As mentioned in section 6.1.2, this topic was also addressed by Paul Graham and Gary Robinson. In their approaches, they introduced measures to estimate the tokens' confidence based upon their occurrences. In equation (6.2) Robinson introduced a way to reduce extreme results for words with low levels of occurrence. Thus, he also applied a kind of confidence measure in a classifier dependent way. This method is also described in [Seg07] and therefore referred to as "segaran" confidence method in this work. Due to the static nature of this approach with its default values, it is only applicable to situations where uncertainty is defined by very small occurrence counts.

As described earlier in paragraph "Support Vector Machines (SVM)" of this section, SVMs are able to measure the confidence of a classification decision based upon the distance of the classified item from the optimal decision boarder. In examples where only SVMs are applied to classification problems, they can be seen as fitting and dependable means of determining the classification confidence.

In our approach we are confronted with different types of classification mechanisms. Thus, a confidence measure of program predictions in a classifier-agnostic way is needed. For this purpose we recommend the use of the inverse variance function of the beta distribution.

$$var[\mu] = \frac{(\alpha\beta)}{(\alpha+\beta)^2(\alpha+\beta+1)} \tag{6.55}$$

Here α corresponds to tokens found in the recommendation and β corresponding to tokens in the rejection class. Both values are scaled with respect to the overall number of tokens and the recommendation-rejection ratio of the current sample. As shown in figure 6.13, the function's value increases with growing values of α and β respectively with the growing number of tokens seen. Between the values 0 and 10 for α and β, the function increases rapidly. Therefore, this interval is most significant for the confidence value. Exceeding a value of 10 the function stabilizes near the value of 1.0. Growing differences in the number of tokens in each class will also lead to a higher confidence. With the given confidence value, it is simple to rearrange the sorting of programs. The sorting is improved by adding a small bonus to more confident and by slightly devaluing less confident predictions from the classifiers' score.

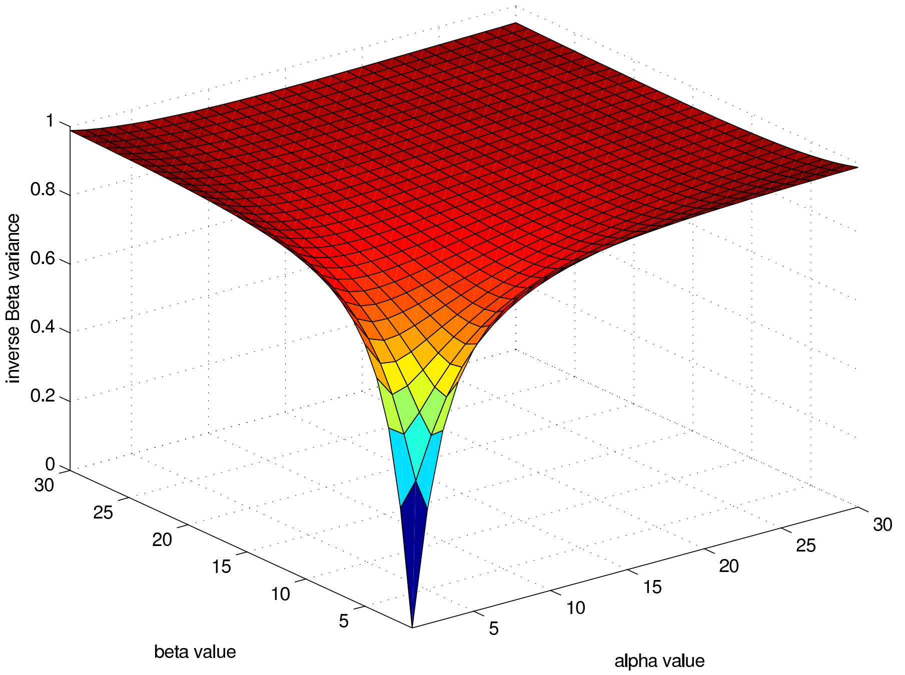


Figure 6.13: Scaled version (factor 12) of the beta distribution's inverse variance function.

All methods and adaptations mentioned in this section have been implemented in our systems and several promising ones will be examined in more detail in the performance evaluation of this work (see section 6.5).

6.3.7 Related Work

Several approaches have been made to provide personalized recommendations to users in a broad variety of contexts. Examples of this can be seen in the areas of educational systems [Got08, Pap06], online forums [CH08] and online shopping systems [Lei07, Gar08]. In our approach, we concentrate on the area of TV program recommendations.

Most systems in this field employ the techniques of collaborative filtering to generate recommendations. One of the most popular approaches that uses collaborative filtering techniques is the TiVo system. TiVo is a complete Personal Video Recorder (PVR) with enhanced recording functions, an EPG-Guide, time shifting functionality etc. To generate recommendations, TiVo accesses a huge amount of content ratings from different users and calculates a similarity measure between them. Favorably rated programs of users that show the most similarities are then recommended.

[Mit10] presents a collaborative approach in which the social context of users is taken into account . This approach makes use of facebook to gather statistics about the channel usage and program usage of different users. Based on this data, the approach provides basic recommendation functions such as the reordering of channels and content lists, and dialogs displaying friend's usage (e.g. ”Two of you friends are watch X on channel Y right now”).

However, recommendation systems using collaborative filtering are plagued by the fact that users may watch similar programs according to one similarity measure but without having common interest apart from that. Hence, collaborative filtering approaches sometimes lead to simply ludicrous recommendations (cf. [Zas02]). Furthermore, a clear advantage of individualized systems over collaborative ones is that they are able to explain results to the user (e.g. in the content-based filtering case by showing the tokens that led to the result) [Stu07].

Other systems use content filtering techniques to generate recommendations. The ”Lightweight Mobile TV Recommender” presented in [Bär08] is a system optimized for mobile devices with DVB-H. It uses content filtering techniques and a preprocessing step to identify topics and emotions from the program descriptions. The system computes recommendations based on queries of the user and the settings entered by him or her for the categories fun, action, thrill and erotic. Compared to our system, this approach permanently needs explicit input of the user (a query and the value settings for fun, action, thrill and erotic). Moreover, it includes no mechanism for automatic adaptation to the user's behavior.

The work of Bjelica [Bje10] presents an interesting approach to applying the vector space model, in combination with pattern recognition techniques, to recommend programs. To determine the correspondence between program vectors and the user profile vector, the Cosine angle between these vectors is used. As we did in our SVM based recommendation approach, we assume that a change in the user's behavior leads to problems in the vector representation of programs (cf. figure 6.28). Furthermore, we do not agree with the argumentation of the author that over-specialization might be a valuable feature instead of a problem.

[Wei08] presents a system for user profile-based personalization in the digital multimedia content field. It makes use of both an implicit and explicit profiling approach by evaluating the user's content consumption. Several weighting factors for single elements and element combinations must be set by the user.

In the approaches of Pronk et al. [Pro10] a recommendation system using naive Bayesian classifiers is presented. Using generated recommendations, a personal TV channel is provided to the users of the system. The practicality of this system has been demonstrated in two showcases - the first on an android smartphone and the second on a typical TV set-top box.

A very early approach to the use of Bayesian filters is described in [Zim04]. It also applies the mechanisms of decision trees and allows for the definition of an explicit user profile. Recommendation fusion of the different recommendation mechanism is done by a neuronal network.

Although several ideas proposed in the papers [Pro10], [Wei08] and [Zim04] seem to be very similar to our approach, they differ in the way they handle recommendation generation e.g. we make use of spam filtering approaches which outperform standard naive Bayesian classifiers. Additionally, our proposal is able to use different classification techniques, which enable it to cope with unstructured metadata. Moreover, relations between different elements are automatically considered. Adopting promising results from spam fighting research, our approach mainly focuses on content-based classification techniques adapted to the multimedia content area. Thus, arbitrary combinations and specializations of recommendation engines can be integrated easily.

The approach of Xu et al. [Xu06] makes use of SVMs to provide personalized recommendations. Compared to our approach the authors solely evaluated the linear and the polynomial Kernel type with a total number of only 6 users. Furthermore, no extensions such as dynamic weight adaptation or ensemble techniques have been used in this approach. To the best of our knowledge Xu's recommender is the only other TV recommendation approach that makes use of SVMs.

In [Got10] the authors present a very promising recommendation approach by applying information retrieval techniques. Compared to our approach, recommendations are generated very differently by using the Okapi BM25 ranking function, which is widely used in search engines. For use in TV recommender systems, this function was extended by a higher weighting of named entities and compound words. We suggest that a combination of our approach and this information retrieval approach might lead to considerable improvements of the recommendation quality of both systems.

The third category of approaches tries to combine content and collaborative filtering approaches in so called "hybrid systems." A very interesting example is presented in [Gud08]. The work aims at personalizing a live program (the Olympic games) by selecting a specific live stream according to the user's preferences, thus avoiding zapping behavior by the consumer. Because of the shortcomings of the collaborative filtering approaches, they adopt the techniques of [Mel01], in case of sparse data, pseudo user ratings must be generated. A metadata-based prediction process based on the user profile fulfills this requirement. In contrast to our more general approach, [Gud08] only focuses on live events. Recommendations are generated based on a rating matrix for the current event. This proposal faces the problem of sparse ratings and must therefore revert to a basis of low quality data to yield recommendations.

6.3.8 Conclusion

This section presents a content-based recommendation engine for TV programs. The system is based on successful lightweight methods adopted from spam fighting research and state-of-the-art classification mechanisms. It proposes a convenient and adaptable recommendation setup that enables the use of multiple classifiers, each specialized in selected elements of the programs' descriptions. An implicit profile derived from the usage history as well as an explicit MPEG-7 based profile, enabling the user to directly influence the system's recommendations, are both taken into account. This creates the possibility of importing other MPEG-7 standard compliant profiles. The system automatically adapts to the evolving user preferences.

The remainder of this section outlines future expansion and enhancements of our content-based recommendation engine.

Interpretation and Weighting of User Actions

The integration of several additional types of user actions (e.g. browsing the EPG, intense zapping, partially watched programs) together with appropriate weights would further extend the capabilities of our recommender. Additionally, introducing a whitelist and blacklist specifying strong likes and dislikes, will give more control to the user, so that his or her explicit preferences overrule potentially conflicting recommendations.

Relevance Feedback

Users should be able to understand, at least to some extent, why a specific program is recommended by the system. That is, the recommendation system must be able to explain it to them. This has the potential to simultaneously enhance the system's popularity and help to improve the recommendation process. Making the evolution process of recommendations clear to the viewer is not that far removed from the relevance feedback approach. The weighting factors of the single evaluation components can be determined by asking the user to rank a set of favored programs and then further adapting them in an iterative process.

Tracking Users' General Interests and Media Consumption

To improve the accuracy and the adaptation process of a recommendation system based on changes in user's preferences, extending the usage history is an interesting option. In order to provide a profound basis for better recommendations, user actions beyond the context of television (e.g. when consuming other media content or surfing the internet) can be taken into account, as well.

6.4 Collaborative Media Recommender

Relying on content-based recommendation methods is often said to suffer from "over-specialization"[Bal97]. This means that these approaches tend to solely recommend items the user has liked in the past or that are at least very similar to such items. This leads to a very limited variation within the recommendations. While content-based filtering in most cases depends on textual descriptors, collaborative filtering (CF) employs ratings of different users and the similarity between them to predict preferences. One advantage to this approach is that it does not require content/metadata about an item in order to perform such predictions. Moreover, it can infer relations between items that have nothing in common except that the same group of users like them. Usually, CF uses a matrix with columns representing users and rows representing items. By either comparing row-vectors (item-based) or column-vectors (user-based) with each other, top-n recommendations and predictions of ratings can be inferred. In tag-based CF, a third dimension is introduced representing the tags. This dimension offers a new possibility to compare users and items with each other. Although some researchers have already focused on this concept [Sen09, TS08, She08], the full potential of recommendation processes including tag information has not been realized.

Thus, the following section introduces a collaborative filtering engine used in personalTV. This component might indeed let the system propose good recommendations that would never have been considered solely based upon a viewing history. In our approach, user-added tags are at the foundation of the system and concepts of the area of collaborative tagging systems are used for recommendation generation. Additionally, a mechanism for inferring new tags for new items (TV programs) is presented. Within our collaborative media recommender component we introduce a combination of methods used in TV guides and folksonomies in an open, self-adapting and personalized recommendation engine. While in many points CF outperforms content-based techniques, it has an important drawback. When a new item joins the system there are no ratings from users and thus, it can not be recommended to anybody until someone rates it (new item problem). Due to the broadcasting structure of the TV realm, trying to apply CF on programs turns out to be a difficult task. Within this system component, we tackled the following issues and present feasible solutions for them:

- **New item problem**
  In most tagging systems only sparse information is available about new items occurring for the first time. Particularly in the realm of TV, new programs are constantly being introduced. Using content-based filtering mechanisms, we are able to infer descriptive information about these items and use them for further steps such as recommendation generation.

- **Individual tags for individual users**
  Tagging often heavily depends on the personal perception of individual users. Even common tags may have different meanings and interpretations depending on the person. In order to take this into account, our approach relies on the local tagging history of each user to suggest tags for his or her new programs. In addition, the system can generate common tags by aggregating individual tag clouds corresponding to different users.

- **Enabling recommendation and tag generation on different levels of participation**
  Even though it contradicts the Web 2.0 philosophy, some users are not willing to share usage histories or even tags with others. Thus, our system enables users to decide how much information they make available while still providing tag and recommendation generation regardless of the chosen setting.

The remainder of this section is organized as follows. First in section 6.4.1 we present the structure of our collaborative recommendation engine. This section additionally covers different user participation modes respecting possible privacy concerns, scalability issues and a high-level description of the engines work flow. Sections 6.4.2 to 6.4.4 detail the main components of the engine. The, section 6.4.4 discusses the most important system part, the tag generation engine (TGE) with its flexible and extensible structure. Section 6.5.1 outlines our test dataset and evaluation methods used. Evaluation results based upon real usage histories are presented in section 6.5.7. In section 6.4.6 we detail several related approaches and discuss how they differ from our proposal. Finally, section 6.4.7 concludes the discussion of our collaborative media recommender with a short summary and a presentation of future work.

6.4.1 Recommendation Engine Overview

Figure 6.14 shows the recommendation engine of our collaborative media recommender component. The engine is partitioned into the client part, commonly deployed directly on the TV set-top box or our personal remote control, and the collaboration server as a central component. Several tagged programs $p_x \in P$ (with P being the set of available programs) building the set H of all tag annotations of individual users are at the foundation of this recommendation system . A tag annotation is represented as a triple (u_z, t_y, p_x) where $u_z \in U$ is the current user and $t_y \in T$ denotes a certain tag associated to p_x.

$$H = \{(u_1, t_1, p_1), \dots, (u_z, t_y, p_x)\}$$

A tag cloud $tc_{(p_x,u_z)}$ describing a program p_x (as shown in figure 6.15) for the user u_z is commonly built by at least one tagging triple and therefore $tc_{(p_x,u_z)} \subseteq H$. The size in the representation corresponds to each tag's weight determined by its frequency. Based on these tag clouds, a Tag Generation Engine (TGE) situated on both the client and the server side, is trained with program metadata. Based on this training data the most suitable tags for upcoming programs are generated. For a detailed description of the TGE, the reader is referred to section 6.4.4. Generated tag clouds are exchanged between server and client and used for the recommendation generation step on both sides. For the final recommendation, the scores from both sides are averaged. Due to the high similarity between the process steps on client and server side the main work flow of the system is detailed in section 6.4.2. Section 6.4.3 introduces additional components used on the collaboration server.

Level of Participation

Although Web 2.0 is based upon the idea of user participation, we do not want to force people to upload personal data in order to get recommendations. In reference to figure 6.14, our approach supports different levels of participation determined by the flow of user specific information and where it is processed:

- **full participation:** The complete user history and generated tag clouds are uploaded to the server. With this mode, the user benefits from all available generation methods of this component and consequently gets the best recommendations. The data will not be made publicly available although the approach of a TV folksonomy gains potential.
- **medium participation:** The user's tag clouds are generated by the TGE, but no tags directly added by the user are uploaded to the server. With this setting, the server can determine neither which programs a user has actually tagged nor how he or she has tagged them. Therefore, inferences of user habits like favorite channel/program, usual watch time or how much a user watches TV can not be easily made. No doubt, a portion of the user's habits also remain in the generated

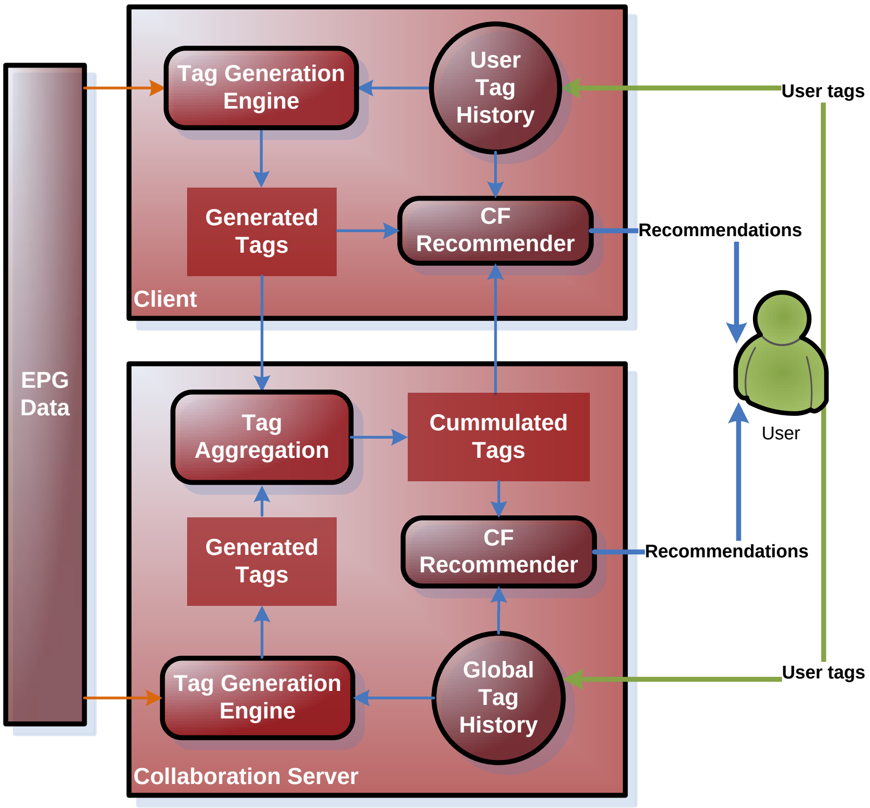


Figure 6.14: Architecture of our collaborative media recommender component.

tag clouds as the TGE is trained, based upon programs previously tagged by the user. On this level, the user will not benefit from recommendation generation on server side.

- **low participation:** This is the most restrictive method in terms of participation. The user uploads no data. He or she only receives the cumulative tag cloud generated by the participation of other users. The tag cloud received by the client is then added to the locally generated one. The recommendation generation suffers from several restrictions and because of this, the quality of the recommendations is assumed to be not as good as with other levels of participation.

In respect to privacy issues, the user is able to decide how much user specific information he or she is willing to share (depending on the mode). It must be acknowledged, however, that if no one is willing to cooperate the whole system degenerates to a solely local and content-based approach.

Scalability

In our approach, load balancing can be achieved in different ways. First, several important steps in the recommendation process are performed locally on the client side, without putting load on the server. Steps such as the TGE or the recommendation generation on the server side can be skipped. Nevertheless, this comes with the price of a lower quality of recommendations. Furthermore, the system can adapt to the amount of available resources on both sides by reducing the number of tags taken into account in the TGE or by limiting the number of learned samples, e.g. by time (cf. section 6.4.4).

6.4.2 Collaboration Server

The collaboration server is realized as a central component where data stemming from all participating users is processed and aggregated. All tags directly added by the user are collected in the collaborative tag history. Based on this history and the associated program data the server-side TGE is shaped using the most popular tags stemming from the tag-aggregation of all users. Thus, it is able to predict commonly used tags for upcoming programs. The resulting tag cloud can be interpreted as an estimation of how the the majority of all users would tag the new program. Commonly, the number of tags in the servers tag generation process can be very high depending on the quantity and diversity of user tags. This tag cloud is used for the construction of the cumulative tag cloud in the tag aggregation step. On the client side, a very similar tag-inference process is also conducted. The client's generated tag cloud is submitted to the server. For each client this tag cloud is calculated by adding up each tag's score to the server's boosted tag cloud (scores are multiplied with by the number of contributing clients). Thus, the resulting cumulated tag-cloud contains tags that are very user specific as well as very common tags. This tag cloud is then distributed to the clients.

Accordingly, the CF recommender measures the similarity between the cumulated tag cloud and the ones in each user's profile. This can be done in various ways and we gratefully refer to section 4.3.1 and to Markines et al. who evaluated different similarity measures for annotated content in [Mar09]. At this point, a typical item-based CF-step is conducted. The similarity scores from the k-nearest neighbors are finally aggregated by calculating the weighted average score (cf. section 4.3.2). As a similarity measure the Pearson correlation as shown in (4.5) has been used. This process, also referred to in this work as recommendation generation, is performed for each new program e.g. the upcoming programs in the next few hours. Thus, we can produce a top-n recommendation of programs for each user.

6.4.3 Collaboration Client

The client is responsible for processing and generating exclusively user-specific and individual data such as tags and recommendations. The metadata used for training exclusively consists of programs originating from a specific user's profile. The TGE on the client side is trained on the user's most popular tags. When a new program is classified by the client's TGE, the resulting tag cloud forms an estimation of how the user would tag the new program. The tag clouds are then transfered from the client side to the server side for further processing. Although no restrictions are made on the selection of tags only a small count of different tags must be considered on the client side (in our dataset, 50 user tags in average). Thus, the clients TGE can also be easily used in resource-constrained environments.

Based on the tag cloud generated using the client and cumulative tag clouds as received by the server, the recommendations are generated by the CF recommender using the

Pearson correlation in a typical k-nearest neighbors step.

6.4.4 Tag Generation Engine (TGE)

With the rising popularity of online services and platforms like Last.fm, Delicious and various blogging systems, users have become familiar with tagging and the way they can use tags for organizing, navigating and discovering items such as images, links, music and videos in huge collections. Tags are generally not restricted by a taxonomy and therefore freely chosen by the user. Thus, finding appropriate and significant tags for an item is a delicate task. In most systems, tag selection is completely left up to the user. Other systems offer selection support by simply proposing frequently used tags. A more sophisticated approach is context sensitive tag suggestion, such as proposing tags that are frequently used in the same topic or category. Another option, as discussed in AutoTag [Mis06], is to suggest tags of other users' articles based on a similarity measure to the article of the current user.

In this section, we describe our approach for annotating new items. It is a fully automated process that utilizes tagging histories for tag generation. In the following section the concept of our tag generation engine is introduced.

The tag generation engine is one of the main components of our recommendation engine. It is responsible for tag inference in the case of the introduction of new programs on the client- as well as on the server side (cf. section 6.4.1). Figure 6.15 shows an overview of this engine's composition. The whole process starts with a set of tagged programs of a user u_z. Each program p_x in the set is described by its tag cloud $tc_{(p_x,u_z)}$. Based on this foundation, a training step is conducted, where binary classifiers are trained using the metadata elements such as title and synopsis (cf. section 6.3.3) associated with each

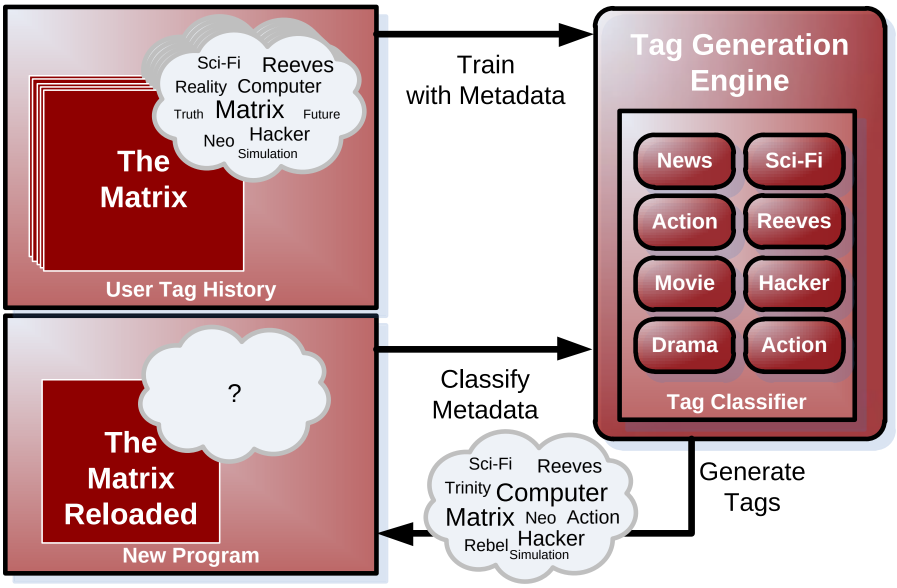


Figure 6.15: The Tag Generation Engine (TGE) and its workflow.

program, is conducted. According to the findings presented in section 6.5 and of [Höl10] we decided to focus on the categories ”Category/Genre”, ”Subtitle”,”Synopsis”, ”Persons” (aggregation of persons occurring as actors, directors, etc.) and ”Title.” Due to the fact that the impact of metadata elements on the user's tagging habits may change over time, (e.g. after finding his or her favorite actors he or she starts to tag movies with actor names) the application of the weight adaptation approaches, as introduced in paragraph ”Dynamic Weight Determination for classifier ensembles” of section 6.3.6, are also preferable in this approach.

For each unique tag t_a a separate classifier ensemble, adapted from the content-based recommendation engine of our system, is used. Through this process it becomes an expert for the single tag and its related metadata. Each tag classifier ensemble is further assembled by a combination of several binary classifiers. For a higher precision in tag inference, and in order to respect restricted resources on devices such as set-top boxes, only tags that occur at least in several triples, currently ten, are used. These tags are noted t_a with $t_a \in S$ and $S \subseteq T$. T stands for the set of all tags. Reducing the number of tags offers a flexible way to control which tags are used in the tag generation step. Thus, settings such as ”only the 100 most frequent tags should be inferred” can be easily implemented. As discussed in section 6.3.3, the selection of appropriate training samples is a sensitive step. In our approach, program metadata related to the current tag (t_a) are used as positive examples (H_{t_a}) in the training of the specific tag-classifier.

$$H_{t_a} = \{(u,t,p) \in H | t = t_a\}$$

For the selection of negative training data, we adopted the ”closed world assumption”: Every tagged program not annotated with the current tag is used for negative training. The data for training of the negative class is therefore defined as the set $H_{\neg t_a}$.

$$H_{\neg t_a} = \{(u,t,p) \in H | t \neq t_a\}$$

It is obvious that the selection of $H_{\neg t_a}$ leads to an imbalance between the number of negative and positive samples. For a readjustment, negative samples are devalued by a weight of $\frac{1}{|H_{\neg t_a}|}$ in the training phase. With the increasing size of profiles, the number of samples taken into account also must be limited to guarantee stable performance of the system. As a result, an iterative training step is conducted for new tag annotations.
Finally, tag generation as a classification task concludes the process. Upcoming programs are classified by each tag classifier. Thus, new tags are added to the program according to the score value of its classifier. In addition to the tag annotation, the score value can be used as an indicator for the strength of the program-tag relation, ranging from 0 (negative example for this tag) to 1 (perfect match). By taking these strength indicators into account and applying a threshold, a *positive* as well as a *negative tag cloud* is easily created to describe the current program. What we call *positive tag clouds* are the same as what is generally referred to as tag clouds: a set of tags describing a program. By contrast, a *negative tag cloud* contains tags that do not describe the program well. These tags can be seen as counterexamples for relevant tags for this program. Negative tag clouds are not directly useful for the users, but for the recommendation system they do offer valuable sources for improving the ranking by further differentiating the programs.

For a detailed description of the classification approach, the reader is referred to section 6.3.3.

6.4.5 Tag Preparation and Enhancements

Due to the fact that tags can be freely selected by the user the dataset contains many different tags. Some tags that are treated as completely different at the moment, are actually closely related (such as ”soccer” and ”soccer match”) or the same words but in different languages (for example, ”cartoon” and ”Zeichentrick,” which is the German translation of ”cartoon”) or even synonymous. Thus, we assume that after a clustering step, tag prediction would be enhanced in both the precision measure and in recall. In our system we make use of a typical hierarchical clustering approach where a hierarchy of tags is build based upon the co-occurrence of tags in tag annotations. In an iterative process, each step merges the two closest groups of tags based upon their co-occurrence until only a single group containing all tags remains. We suggest that utilizing a semantic distance function can further improve clustering. Based on a measure like the Leacock-Chodrow Measure defined in section 3.4.2, the relation of tags modeled in wordnet or GermaNet can also be incorporated into the distance calculation of tags and tag groups.

6.4.6 Related Work

Before we turn to the discussion of related work, it is important to note that much of it focuses on datasets which are very different from those of the TV realm. However, our approach relates to Social Tagging and Recommender Systems. We can roughly distinguish three related areas: tag prediction for resources using content/metadata, additional tag prediction for resources based on few assigned tags and tag-based recommender systems.

The idea to generate tags using the content of a text (in our case metadata) is not new. Heymann et al. [Hey08b] introduce SVMs to classify different descriptors of web pages such as content, anchor texts and information about surrounding hosts. Measures on the Stanford Tag Crawl Dataset [Hey08a] prove that classification can provide very promising results. Unfortunately, the questions about performance and scalability of the demonstrated method are left unanswered. The approach of AutoTag [Mis06] uses well known Information Retrieval measures to recommend tags to a user who creates a Weblog post. Posts which are similar to the active post are retrieved by using a ”distinctive term” query. A set of tags exists which is assigned to the most similar posts. Each tag from this set is then ranked by its frequency of occurrence. User preferences are also taken into account: tags from the previously mentioned set are boosted by a constant factor if the blogger has already used them before. In [Zha09] the authors introduce a combination of the Language Model [Zha04] and the ACT model [Tan08] to handle the new item and the new user problem of CF. This combination results in f-measure scores below 15%, so improvements are necessary.

A graph-based approach to the recommendation of tags in folksonomies is described in [Jäs07]. The method in use is called FolkRank [Hot06], it is based on the idea of the famous PageRank algorithm. In this theory, rankings of resources are determined by the assumption that important resources are labeled with important tags by important users.

Another discipline in tag recommendation focuses on predicting additional tags for resources which are already annotated with a few tags. In [Hey08b] the authors demonstrate how association rules can be used to predict additional tags. The rules constitute relations like ”osx → mac.” This approach is outperformed by the method from Krestel et al. [Kre09] which is based on Latent Dirichlet Allocation (LDA). Sigurbjörnsson and van Zwol [Sig08] use the dataset of Flickr to exemplify how tags can be recommended by analyzing co-occurrences. A related approach introduced by Wu et al. [Wu09] not only uses co-occurrence but also visual correlation. The effectiveness of this multi-modal system is also demonstrated on the Flickr dataset.
Another set of algorithms is referred to as ”tag-based recommender systems.” In contrast to tag recommender systems, these systems utilize user annotations for resources to infer user preferences. Sen et al. [Sen09] combine a user’s tag- and movie preferences to provide CF-based recommendations for other movies a user might like. The authors demonstrate that a combination of tags and numerical ratings performs better than purely tag-based methods. Similarly, a CF-based approach is also used in [TS08], in which the researchers introduce a tag-based CF algorithm that combines two CF methods, namely user-based CF (user+tags) and item-based CF (items + tags). The evaluation on Last.fm data shows promising results. Another method used by Shepisten et al. [She08] first calculates the cosine similarity between tags and resources. Additionally, tags are clustered off-line and a user’s interest in each cluster is determined. Eventually, personalization of the initially estimated list of recommended resources is performed by including the similarity of a cluster to a user’s preferences.

6.4.7 Conclusion

Tagging is gaining more and more attention in various application areas. Especially in the context of Web 2.0, it is a well established method for simplifying organization, navigation and exploration of large collections of items. In particular, several recommendation approaches based on tags have been successfully applied. This section presents our unique approach to overcome the main limitation of most of these systems when new items (in our case, programs) are introduced. Based on a content-based filtering approach, we present a individualized and flexible solution for tag generation allowing different users to participate at different levels. This ranges from no cooperation to the full sharing of all tags and profiles. Based on generated and user given tags, a collaborative filtering method is applied to compute recommendations. The remainder of this section outlines several future expansions and enhancements.

Employing bigger datasets
Our evaluation results clearly indicate that with more training data, the relation between tag and metadata could be more precisely revealed. Moreover, the power to discriminate between different tags is elevated with increased training. Thus, we are planing to utilize the MovieLens[1] dataset augmented with movie metadata available from movie databases such as the IMDb.

[1] Movielens - http://www.movielens.org/

Taxonomy generation
Based on the tag clouds which are collected on the server, one can discover different relationships between tags. The most important relationships for recommendation is of the type ”is-a.” An example could be that we have the tag ”football” for a program but not the tag ”sports” although people who are interested in sports might be interested in football as well. To discover and use such relationships we plan to evaluate a method based on WordNet[1].

User specific metadata selection
As stated in section 6.4.4, we believe that the user’s tag selection for a program is not influenced by all metadata elements equally. One user’s tag selection might be mainly driven by the genre or by specific actors, whereas for other users, the synopsis might be of higher importance. By observing the users behavior and revealing such relationships, a proper selection and adaptation of metadata could be realized faster and more accurately than with the dynamic weight adaptation mechanisms discussed in section 6.3.6.

Better selection of negative training samples Our system currently uses all other tags and their related programs as negative examples for a given tag in the trainings phase. Although this method seems to work quite well, (cf. the evaluation results in section 6.5.7) in some situation it will hurt the performance of the system. Especially co-occurrent tags could lead to the inclusion of a program in the positive samples as well as in the negative samples. Based upon the results of a co-occurrence analysis and a clustering step, this situation could be easily avoided.

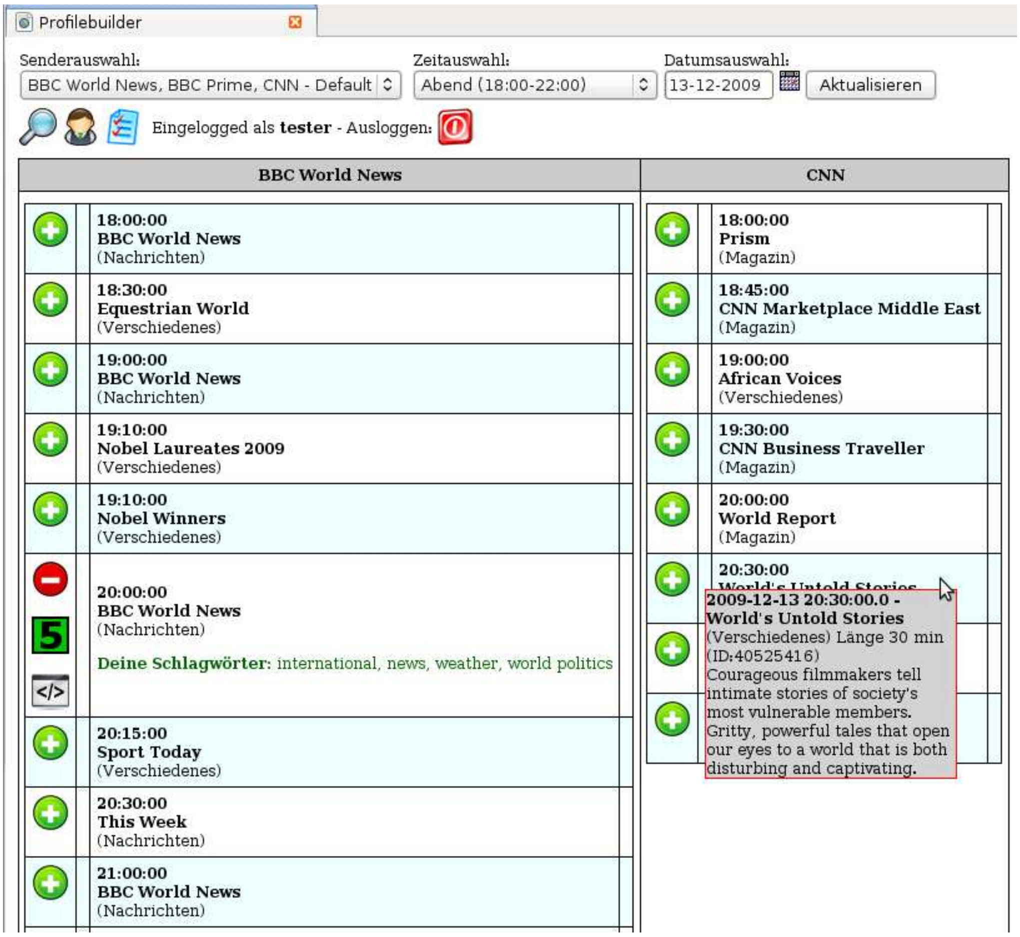
6.5 Experimental Results and Evaluation

In most cases, the quality of recommendations is a subjective matter that depends on the user’s individual point of view. Thus, measuring the performance of such systems is a challenging task that often requires a trade-off between the results’ expressiveness and their objectivity. Therefore, a use-oriented test setup with specific indications must be defined. The following section is organized as follows: First, in section 6.5.1 we introduce the structure of our test dataset and how this dataset was created. Section 6.5.2 details the methods and measures used for evaluating our system. Besides well established measures from the literature, several concepts designed within the evaluation phase of our system will also be presented. Based on this methodological foundation, section 6.5.3 to section 6.5.6 presents and analyses the evaluation results of our content-based engine. Finally section 6.5.7 of our collaborative recommendation engine.

6.5.1 Creation and Structure of the Test Dataset

As there is no publicly available EPG-dataset applicable in the context of our work, we have decided to collect our own dataset. More than 100 users were asked to participate in

[1] WordNet - http://wordnet.princeton.edu/



Profilebuilder

Senderauswahl: BBC World News, BBC Prime, CNN - Default

Zeitauswahl: Abend (18:00-22:00)

Datumsauswahl: 13-12-2009 Aktualisieren

Eingelogged als **tester** - Ausloggen:

BBC World News	CNN
18:00:00 **BBC World News** (Nachrichten)	18:00:00 **Prism** (Magazin)
18:30:00 **Equestrian World** (Verschiedenes)	18:45:00 **CNN Marketplace Middle East** (Magazin)
19:00:00 **BBC World News** (Nachrichten)	19:00:00 **African Voices** (Verschiedenes)
19:10:00 **Nobel Laureates 2009** (Verschiedenes)	19:30:00 **CNN Business Traveller** (Magazin)
19:10:00 **Nobel Winners** (Verschiedenes)	20:00:00 **World Report** (Magazin)
5 20:00:00 **BBC World News** (Nachrichten) **Deine Schlagwörter:** international, news, weather, world politics	20:30:00 **World's Untold Stories**
20:15:00 **Sport Today** (Verschiedenes)	
20:30:00 **This Week** (Nachrichten)	
21:00:00 **BBC World News** (Nachrichten)	

2009-12-13 20:30:00.0 - World's Untold Stories
(Verschiedenes) Länge 30 min
(ID:40525416)
Courageous filmmakers tell intimate stories of society's most vulnerable members. Gritty, powerful tales that open our eyes to a world that is both disturbing and captivating.

Figure 6.16: Screenshot of the ProfileBuilder's web interface.

our survey. In the survey, program information for a total number of 158 TV channels was available. For the data acquisition, we developed a simple web application named ”ProfileBuilder” (see figure 6.16). Users were asked to add all the programs they have watched to their histories (by clicking on the plus sign). Programs were presented in a tabular program overview. For each program, detailed metadata such as synopsis, subtitle, etc. were presented as a tooltip. The users were able to search for programs in a specific period of time by title or in a full text manner. To improve usability, configuration options such as the selection of favorite channels for the program overview were available. Furthermore, the users were able to assign tags to programs in their viewing history. Tag selection was not restricted, neither in the vocabulary nor in the amount of tags. To avoid errors, the user was also able to remove programs from his or her viewing history and to edit and delete his or her tags related to a program.

Over a period of 10 months, a total number of 67 user participated and we were able to collect TV viewing profiles with a total of 10,845 programs broadcast on 66 different TV channels. Figure 6.17 shows the distribution of these programs and user profiles.

On average, each user history contains about 162 programs with the highest program count of 1300 and the lowest of 2. More than 50 % of our user profiles contain 100 or

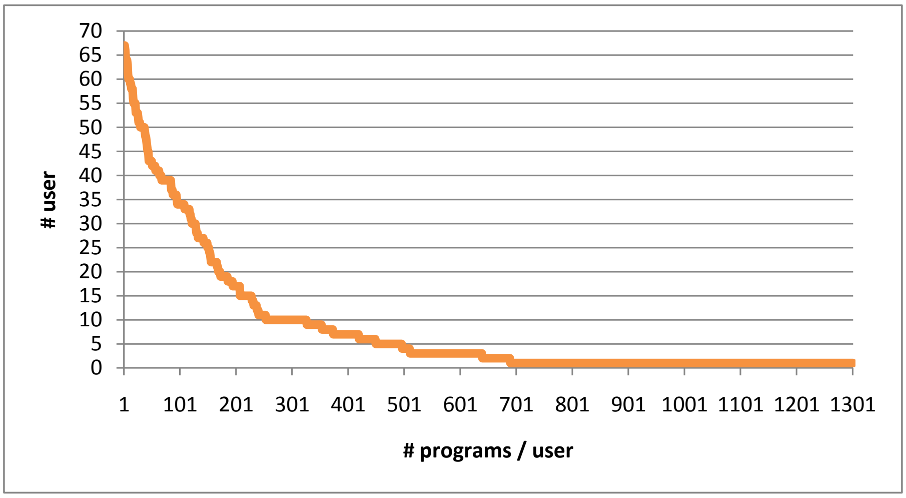


Figure 6.17: Number of programs per user profile in the evaluation dataset.

TV channel	Number of occurrences
ProSieben	2,484
ORF 1	1,649
RTL	936
ARD	772
Kabel 1	636
ORF 2	482
Sat 1	451
Vox	401
ZDF	377
BR	377

Table 6.4: The ten most frequent channels in the evaluation dataset.

more programs. Table 6.4 shows the 10 most popular channels of our users. Our tag dataset currently contains 4515 tag assignments to 1693 different, unique programs. The tag annotation has been performed by 31 active users. The distribution of the tags in our dataset follows a Zipf distribution (see figure 6.18), which is typical for many man-made and natural phenomena. Please note that the most frequent tags are not necessarily the most descriptive ones. The tagging behavior strongly varies between users: while some people use very personal tags like ”want to see again” others choose tags which can be found in the EPG data like the genre of the program.

Nearly all available genres such as science fiction, comedy, thriller, etc. were covered by the profiles. The profiles, chronologically ordered, serve as input as well as benchmark for the system’s performance evaluation. Each profile is used in an iterative training followed by a classification step that evaluates the test parts of the user history. The score calculated is further interpreted as an indicator of the recommender’s ability to approximate the users’ viewing habits.

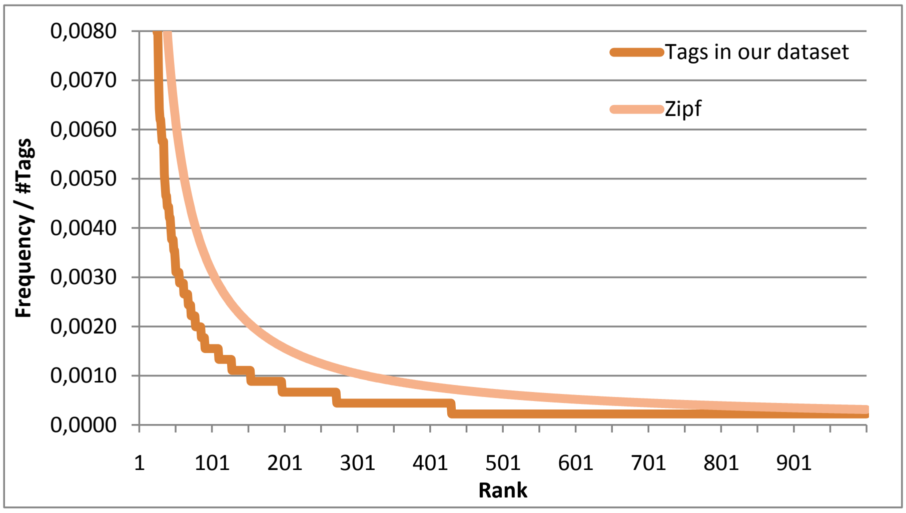


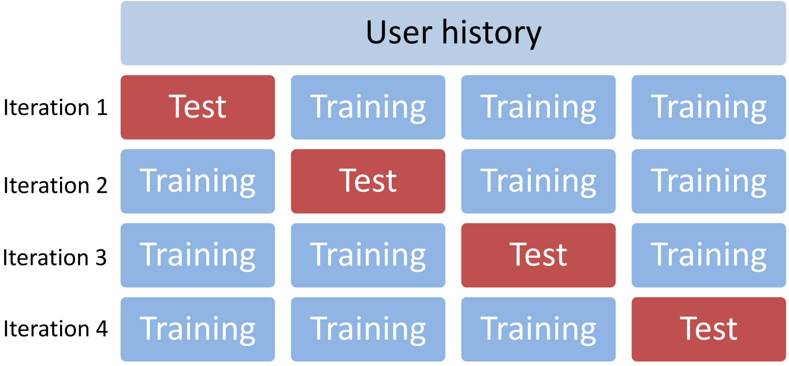
Figure 6.18: Distribution of tags in the evaluation dataset.

6.5.2 Evaluation Methodology

For evaluating the system's performance in terms of accuracy and ability to adapt to a user's preferences, we used different evaluation measures. Aside from common measures used in the fields of information retrieval and recommender systems, such as precision, recall and f-measure, other methods like average rank among concurrent programs have been used in our approach. As a common test strategy, we adopted the cross-validation approach. Cross-validation is widely used for estimating how well a prediction model, trained using a portion of a dataset, is going to perform once put into practice with a more general, independent dataset. Due to the broadcast nature of TV, the dataset is arranged chronological in our work. The cross-validation process can be described as follows: First, the dataset is split into at least two non-overlapping parts. Secondly, the prediction model is trained based using one of the parts. The validation is done by classifying the second part, also called test or evaluation part. Afterwards, training and test part are exchanged. Generally, a n-fold-cross-validation is used, in which the dataset is split into n continuous parts in a random way. The validation is done in n iterations where all parts are used once as a test group. The overall result is calculated as the average of all n separate tests. Figure 6.19 exemplifies this process for a 4-fold-cross-validation. The 4-fold setting has also been used in most sections of the evaluation. In order to represent the nature of our system's use case as accurately as possible, we have used an evaluation setting in which only the last parts of the viewing histories are used. In this setting no interchanging of training and test parts is done.

In the following, the measures used in the evaluation will be briefly introduced:

- Precision and recall: These measures were first introduced in the information retrieval domain, and are among the most frequently used measures. Both measures are also commonly used for evaluating the performance of recommender systems. Generally speaking, the precision value measures the fraction of relevant items correctly recommended, out of all recommended items. The recall indicates how much of the



	User history			
Iteration 1	Test	Training	Training	Training
Iteration 2	Training	Test	Training	Training
Iteration 3	Training	Training	Test	Training
Iteration 4	Training	Training	Training	Test

Figure 6.19: The 4-fold-cross-validation process.

relevant items where recommended out of all relevant items in the dataset.

$$precision = \frac{\#\ relevant\ items\ recommended}{\#\ recommended\ items} \tag{6.56}$$

$$recall = \frac{\#\ relevant\ items\ recommended}{\#\ relevant\ items} \tag{6.57}$$

Both values are within the interval $[0.0, 1.0]$ where 1.0 indicates the best achievable result. Applied to our domain, the relevant items are all of those listed in the user's viewing history. Recall and precision are closely related. As a result, both values need to be considered in order to produce a proper performance evaluation.

- Precision and recall at k: Particularly in web searches, the user is often interested in how many applicable items there are on the first page of results. The size of the result page varies and is defined by a fixed number (k) of results. This measure differs from the common recall measure in the number of items taken into consideration. Here, the relevant items are limited to the k most important items. Equal to precision at k, Mishine [Mis06] defined recall at k.

- F-Measure: As a single value, f-measure takes into consideration both precision and recall. The f-measure can be seen as a trade-off between both values. It is defined as the weighted harmonic mean of precision and recall. The term "f-measure" often refers to a balanced f-measure where precision and recall are equally weighted:

$$f - measure = \frac{2\ precsion\ recall}{precsion + recall} \tag{6.58}$$

- Score: One of the most basic measures is the average classification score and its variance . Generally, a score near 1.0 is considered to be the best possible result. Please note that the average score of a faulty classifier which assigns 1.0 to every item is also 1.0 with a variance of 0.0. Thus, this measure must be considered in combination with other measures in order to produce a proper evaluation result.

- Rank: This is particularly pertinent in situations where recommendations are presented to a user as a list of the most promising recommendations sorted by classification score. In such a system, the most interesting items should obviously be ranked as high as possible. Thus, the ranking of a specific item among a set of other available items can be used as an evaluation measure. In our case the ranking of a specific TV program, as compared to other concurrent programs in a specific time window is evaluated.

As the viewing history only states positive user actions, we needed to find a way to include samples for the rejection class. We resolved this by stating that all programs simultaneously broadcast with an effectively watched program are considered to be rejected. Thus, they are used as negative examples with a very low weighting value (cf. section 6.3.3). Note that a lower weighting of rejections is due to limitations of the mechanisms solely possible with the Spam filtering approaches. Programs watched are learned, fully valued, as ”recommendations.” Several tests with different classifier configurations have been carried out. A selection of the most significant results is presented in this section. All evaluation runs and tests presented were conducted on a high performance cluster. This machine has a total number of 16 computation nodes and one master node. Each node is equipped with two Intel Xeon Quad Core processors with 2 GHz clock rate, 16 gigabyte RAM and two SATA hard disks with 74 GB capacity each. All tests were done in a single threaded manner.

6.5.3 Selection and Setting of Classifier Mechanisms

Most classification methods differ considerably in processing time, resource consumption, recommendation quality and complexity. In the following evaluation we will focus on those mechanisms discussed in section 6.1. Each mechanism offers different techniques for classifying individual items and a variety of parameters to optimize the classification process. Thus, each mechanism was tested in several different configurations. To improve and ease the classification setting, we developed an XML schema for setting up classifier ensembles (cf. section 6.3.3) in an easy and flexible way. Figure 6.20 shows an excerpt of this schema. Each classifier ensemble is made up by at least one specific classifier. Theoretically, an unbound number of different classifiers can be configured. Each individual classifier is identified by its name attribute. This name should be unique among all classifiers used simultaneously in the system. The metadata selection specifies which metadata elements of the program descriptions are used in the classification and learning steps of the classifier. In a typical setting, a classifier is specialized in a single metadata element. Nevertheless, for special cases such as strongly resource-constrained environments or closely related metadata elements (e.g. title and subtitle) the assignment of several metadata elements to a single classifier may be preferable. Each metadata element can be further configured by its individual attributes. Typically these settings are made on a higher level for the whole classifier. The following options are available:

- **addPrefix:** This setting allows for the use of one or more specific metadata elements as the prefix for a token. For instance, a specific token of the title element may be prefixed with title itself (e.g. ”title@Dr. House”) or with multiple metadata elements such as title, channel and genre (e.g. ”title@Pro7@Series@Dr. House”). This prefix can be used to specify the context of a token and do the classification step in a context sensitive way. The usage context (e.g. a specific device in use or the current

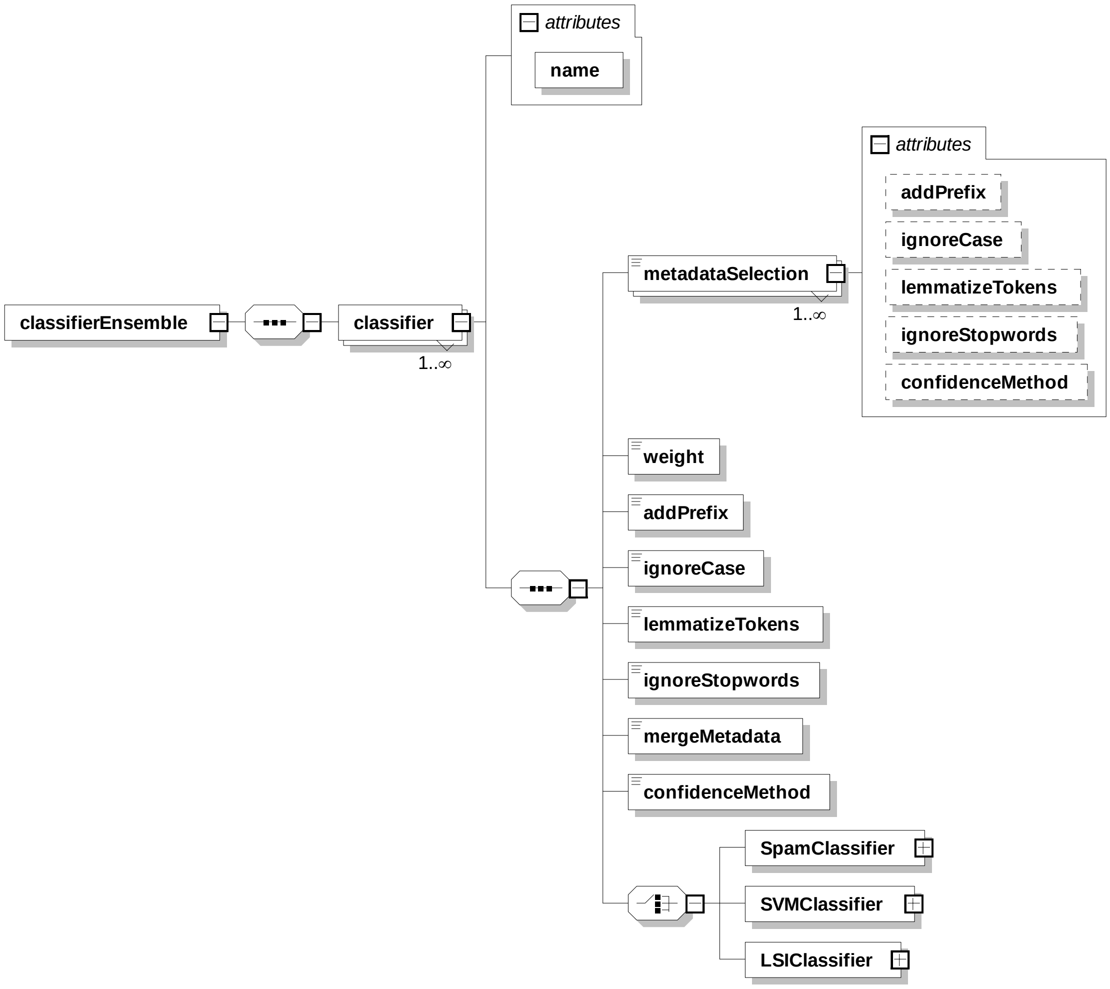


Figure 6.20: The overall classifier ensemble XML Schema.

location) as discussed in section 5.1 may also be realized with the help of adequate prefixes.

- **ignoreCase:** This element specifies whether the processing of tokens should be done in a case sensitive or insensitive way.
- **lemmatizeTokens:** If the lemmatizeToken is set to the value true, only lemmata of all tokens are used.
- **ignoreStopwords:** This setting allows for the omission of all tokens that are considered to be stopwords. These stopwords are identified by matching tokens and different, language specific stopword lists.
- **mergeMetadata:** Generally, different metadata selections are separately processed by each classifier. If mergeMetadata is set to true, tokens of different metadata elements will be processed as a combined set regardless of their original metadata element.

- **confidenceMethod:** This element specifies if and which method should be used to measure the confidence of a certain token and the related classification result (cf. paragraph ”Confidence Value” of section 6.3.6). Currently this setting is only used with the Spam classification setting.

Settings on the classifier level are valid for all metadata selections of this classifier. Nevertheless, it is possible to configure each metadata selection in an individual way. The weight element of the classifier defines the impact of its results on the overall outcome of the whole ensemble. All individual classifier weights are scaled to a percentage between 0 and 100 %. In the case of classifiers with multiple selected metadata elements, tokens from the different elements are equally weighted in the calculation of the classifier’s result. Finally, to complete the configuration of a classifier, the type of classification mechanism used must be selected. The system offers three different mechanisms: the Spam, the SVM and the LSI classifier.

The following paragraphs focus on each of the individual classification mechanisms and evaluates the best classifier setting. First each paragraph focuses on the XML Schema of the individual classification mechanisms. Then the test settings are presented. Finally, we present the evaluation results of each configuration option. To distinguish between different configurations of the classifier mechanisms, we examine the ranking of the different settings per user viewing history (as the ”**ratio of top settings**”) in a 4-fold-cross-validation step. As measures, we used the average program rank among all concurrent programs and the average classification score of these programs. Concurrent programs were selected from all 66 different TV channels included in our viewing histories. A total number of 60 viewing histories, each with a minimum of 15 programs, was used in each test.

Note that the evaluation results presented in the following paragraphs allow solely for the determination of the best overall setting of each classification mechanism. No comparisons between different classification mechanisms may be drawn based on these results.

Spam Filtering Approaches

This paragraph evaluates the different settings of our Spam filtering classification approach. A detailed description of this approach may be found in sections 6.1.2 and 6.3.6. Figure 6.21 shows the XML Schema of the Spam classifier configuration. The schema allows the setting of the following attributes:

- **classifierType:** Valid values for this setting are the three major classification approaches: the Graham, the Robinson Geometric Means (RGM) and the Robinson Fisher (RF) Methods.

- **assumedProbability:** The assumed probability is used to model the ratio of positive and negative samples in the distribution. 0.5 is commonly used as a neutral value for unbiased classification.

- **hapaxesHandlingMethod:** This setting determines the treatment of unknown or rarely occurring tokens. As discussed in section 6.1.2 hapaxes may be ignored (”Ignore_Hapaxes”) or the assumed probability may be assigned (”Ignore_hapaxes_ smooth_probability”).

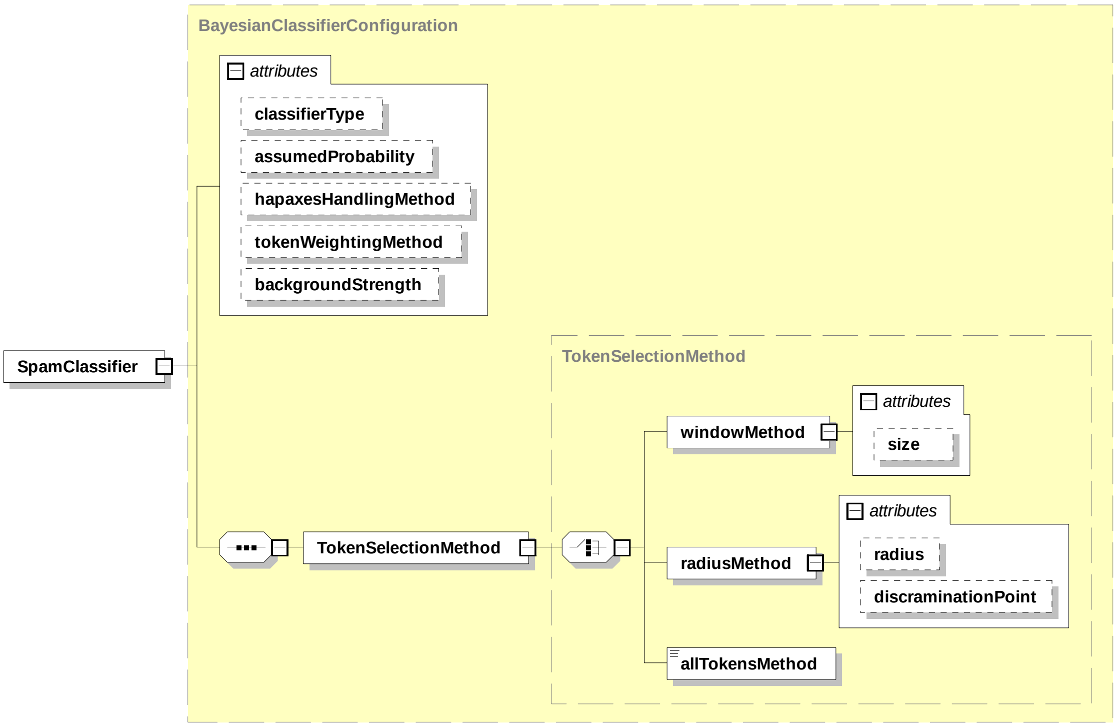


Figure 6.21: The Spam filter based configuration XML Schema.

- **tokenWeightingMethod:** As tokens can occur multiple times in one sample, different ways to deal with them have been proposed. Either they can be counted only once ("Single_count"), a small value can be added to the tokens score ("Single_count_with_bonus_for_multiplies") or they can be considered multiple times in the probability calculation ("Multiple_count").

- **backgroundInfo:** This setting models the sensitivity of the classifier with respect to changes within the user's habits (cf. equation (6.2)). Typically it is set to 1.0.

Additionally, the selection method of decisive tokens can be configured. Valid settings are the "window", the "radius" and the "allTokens" methods. The "window" method allows the user to set a specific window size, which determines the number of tokens considered in the statistical combination step. The "radius" mode uses a radius and a defined neutral point ("discraminationPoint") for token selection. In the "allTokensMethod" setting, no specific filtering criterion is applied and all available tokens are selected.

In order to compare different classifier settings, a comprehensive evaluation with more than 3,300 different classifier ensemble configurations has been conducted. To construct these configurations, all valid combinations of the settings shown in table 6.5 have been tested. Values for the window size and radius value were chosen according to the results of preliminary tests.

We compare the results obtained by the classifiers presented in section 6.1: Graham, RobinsonGeometric and RobinsonFisher. Figure 6.22 shows the results of our comparison of classifier types. Among the three different classifier types, the Robinson Fisher classifier

Setting	Value
Metadata element	"Title", "Director", "Role", "Persons", "ActorsAndRole", "Category", "Subtitle", "Synopsis", "Theme", "ChannelId", "Description", "Country"
Cassifier type	Graham, Robinson Geometric Means, Robinson Fisher
Token weighting method	"Single_count", "Single_count_with_bonus_for_multiplies", "Multiple_count"
Hapaxes handling method	"Ignore_Hapaxes", "Ignore_hapaxes_smooth_probability"
Token selection method	window, radius, allTokens
Confidence method	none, beta, segaran
Ignore case	true
Ignore stopwords	true
Window size	1,5,10,15,20,25,30,40,45,50
Radius values	0.1,0.15,0.2,0.25,0.3,0.35,0.4

Table 6.5: Test settings of the Spam filtering approach.

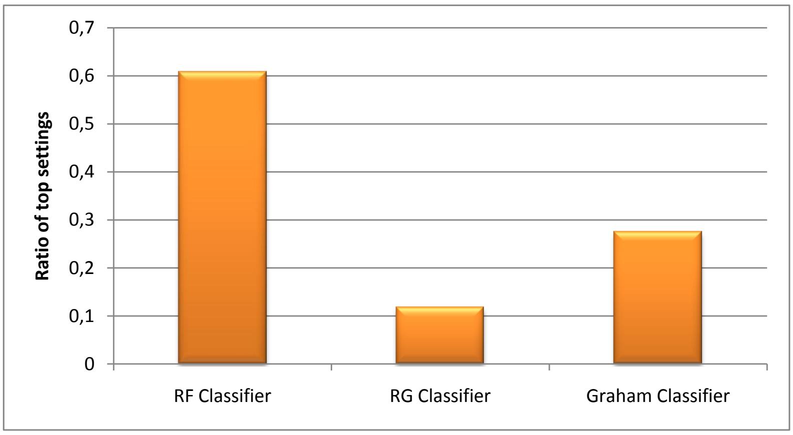


Figure 6.22: Overall ranking of the Spam filter classifier types.

shows superior performance. It performs best for the 60 viewing histories in more than 61% of all test configurations. Examining the test results, it is clear that the performances of both of the other classifier types are less promising. The Graham classifier scores 27% and the Robinson Geometric Means classifier only 12%. Figure 6.23 shows an excerpt of the same comparison grouped by the metadata elements used. Here the differences between the classifier types are not as obvious as in figure 6.22. Nevertheless, the Robinson Fisher classifier still performs best for most metadata elements. In three cases the Graham classifier shows a slightly better result than the RF classifier, whereas the RGM classifier is never the best.

Figure 6.24 presents a comparison of our three settings for the confidence method. It shows that among these methods the Beta confidence calculation (cf. paragraph "Confidence Values" of section 6.3.6) performs best with a percentage of 78%. Another outcome of this figure is that using a confidence method is always preferable because both

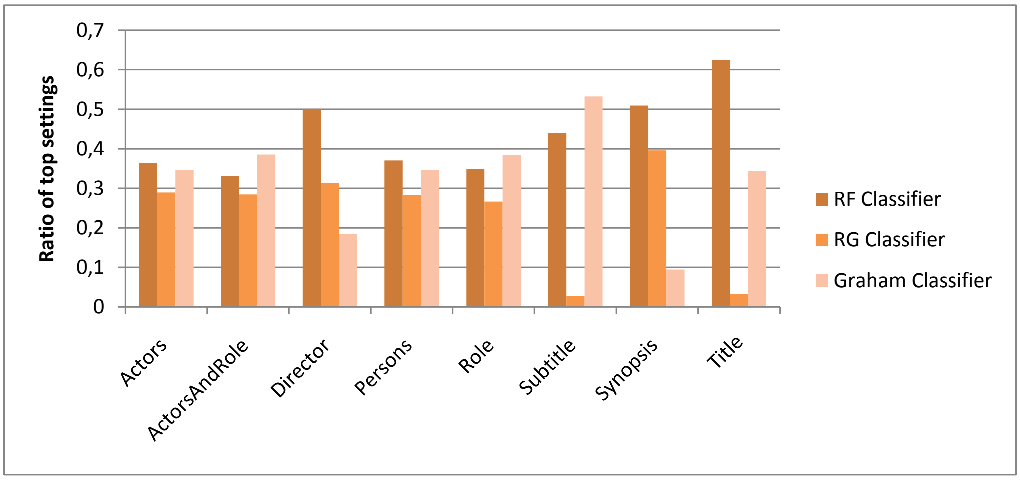


Figure 6.23: Ranking of the Spam filter classifier types per metadata element.

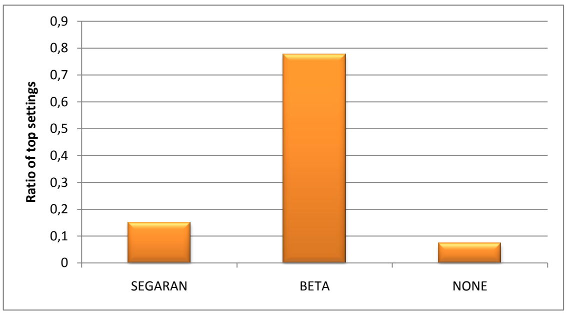


Figure 6.24: Confidence method comparison.

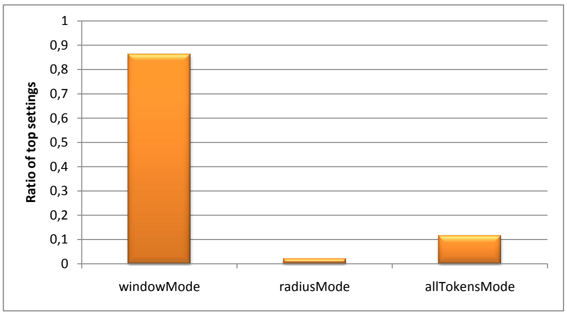


Figure 6.25: Selection method comparison.

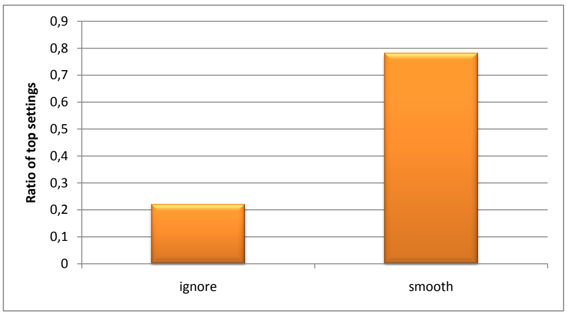


Figure 6.26: Hapaxes handling method comparison.

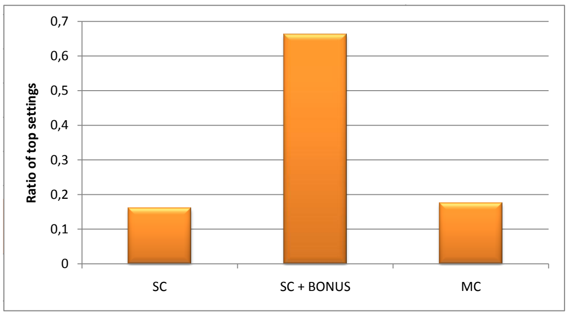


Figure 6.27: Token weighting method comparison.

confidence methods have a higher ratio than using no confidence calculation.

The comparison of our token selection methods is presented in figure 6.25. Here the ”window” mode features the best ratio with 86%. Both of the other methods are negligible with a ratio of 2% for the ”radius” and 12% for the ”allTokens” mode. A detailed examination of the different window sizes revealed that settings with sizes between 10 and 50 tokens in the selection window scored nearly the same, each about 10%. The best ratio was reached at a size of 40 tokens.

Both results, the good performance of the Beta confidence method and the window method, are indicators of the presence of many tokens where no clear assignment to one specific class can be made. The window method typically performs better than all other methods in situations where a small amount of decisive tokens is among a huge amount of tokens, whose classifications are very vague.

The hapaxes handling methods have been reviewed in figure 6.26. It shows that the ”Ignore_hapaxes_smooth_probability” clearly outperforms the ”Ignore_hapaxes” setting with 78% versus 22%.

For the token weighting methods, our tests (cf. figure 6.27) showed the best performance for the ”Single_count_with_bonus_for_multiplies” (66%). The other two methods feature a similar low ratios (16% ”Single_count” versus 17% ”Multiple_count”).

As an overall result of this paragraph we were able to identify the best classifier ensemble configuration as follows: The Robinson Fisher classifier with the Beta confidence method, the window token selection method with a window size of 40, smoothing of hapaxes and the ”Single_count_with_bonus_for_multiplies” token weighting method.

Support Vector Machine Approaches

This paragraph evaluates the different settings of our SVM classification approach. In order to apply SVMs to the task of classifying TV programs, a fixed numerical vector form for representing our samples must be found. In the ”Support Vector Machine (SVM)” paragraph of section 6.3.6, we discuss two ways of collecting fundamental documents for the construction of this feature vector. The ”common document collection” uses all available documents and is able to provide a common vector form for all users. By contrast, the ”user specific document collection” uses only the documents of a user’s viewing history and must be built for each user separately. Thus, we need to decide which vector construction to use. In a preliminary evaluation we constructed a vector using all of the programs of one single month in order to evaluate the feasibility of such a vector. Note, that this vector form would only be able to represent the training samples of this month in a satisfying manner. As a result, frequent reconstructions of this vector are needed to keep track of changes of the user’s viewing habits. Table 6.6 shows the number of dimensions, for a subset of the metadata elements that are supposed to have a high number of different features, needed to represent all programs of this time period. Please note that the high dimensionality also negatively impacts runtime and memory consumption (especially in resource-constrained environments).

In figure 6.28 we examine the stability of such a vector for the metadata elements ”title” and ”synopsis”. After only seven days without updating the vector form, more than 10% of all features could no longer be represented by the vector form. The observation of the weekly and very regular rates of loss are also of interest. These loss are a triggered by

Metadata element	Number of features
Synopsis	160.721
Title	14.256
Subtitle	24.558
Actors	28.412

Table 6.6: Vector size ”common document collection”.

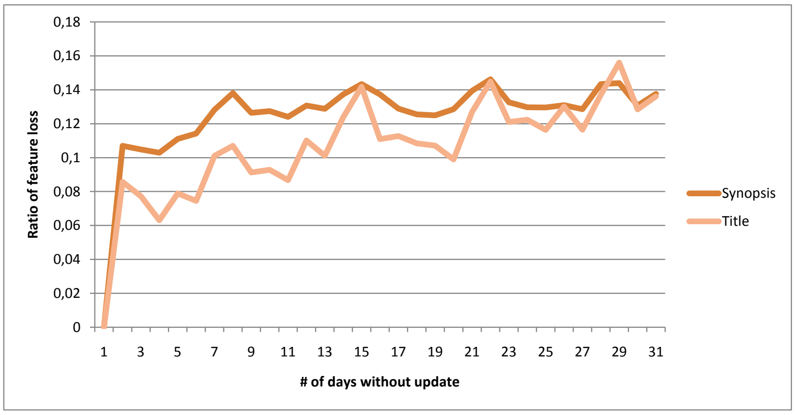


Figure 6.28: Feature loss of a ”common” vector form.

the beginning of new weekly TV programs and series. The course of the loss ratios rise continuously until the end of the evaluation period, with a maximum loss ratio of 15.6% for title and 14.6% for synopsis.

Because of the high dimensionality of the metadata elements (cf. table 6.6) and the ratio of feature loss (cf. figure 6.28), the application of a common vector form is not recommended in our case. Thus, in the following evaluation we make use of a user specific vector form.

For a detailed description of our SVM approach, the reader may refer to sections 6.1.3 and 6.3.6. Figure 6.29 shows the XML Schema of the SVM classifier configuration. The schema allows for the setting of the following attributes:

- **rocSVM:** If RocSVM is set to the value true, negative training samples are extracted using the Rocchio SVM (RocSVM) mechanism. Otherwise all concurrent programs from different TV channels mentioned in the user’s viewing history will be used to evaluate each program.

- **termMode:** This setting determines the method of extracting and representing terms. Possible settings are raw frequency (RF), term frequency (TF) and term frequency inverse document frequency (TF-IDF). A detailed discussion of these settings can be found in the ”Support Vector Machines (SVM)” paragraph of section 6.3.6.

- **c:** C specifies the costs related to a violation of the hyperplanes and the decision border of the soft margin SVM.

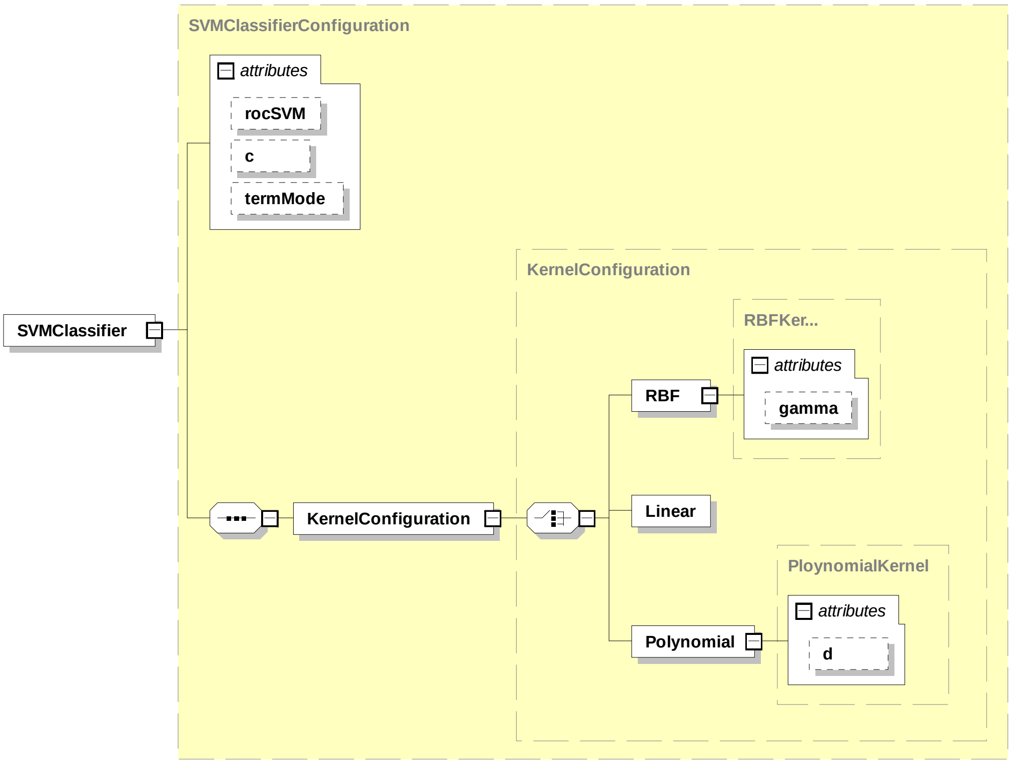


Figure 6.29: The SVM based configuration XML Schema.

The ”KernelConfiguration” element allows for the selection of a specific SVM kernel for the classifier. Currently the radial base function (RBF), the linear and the polynomial kernel are supported. Additionally, the RBF kernel provides the attribute ”gamma” and the Polynomial kernel ”d” for setting the polynomial degree.

To determine the best SVM classifier ensemble configuration, a total of 2,100 different classifier ensemble configurations have been tested. These configurations were constructed by choosing all valid combinations of the settings shown in table 6.7.

Figure 6.30 presents a comparison of our different kernel types. It is clear that the RBF Kernel outperforms the Linear and the Polynomial Kernel by far. In 87% of all configurations the RBF kernel performs best on our viewing histories compared to 6% for the Linear and 7% for the Polynomial Kernel. This result attests to the fact that the classes of TV programs are not easy to separate in vector space. A very similar result is shown in figure 6.31. It shows an excerpt of the kernel type comparison grouped by the metadata elements used. Again, the RBF Kernel outperforms both other kernel types by far. The Linear and the Polynomial Kernels show comparable performance.

After taking the results shown in figures 6.30 and 6.31 into consideration, we decided to concentrate on the RBF Kernel in all further tests. Using the RBF Kernel we conducted a sparse grid search to find adequate values for settings C and $Gamma$. According to the findings of [Hsu03] we selected C and $Gamma$ out of the set $\{2^{-8}$, 2^{-5}, 2^{-3}, 2^{-2}, 2^{-1}, $0.75, 2^0$, 2^1, $2^3, 2^5\}$. In accordance with the results of preliminary tests, more values near 0 were evaluated. Figure 6.32 shows the results of the grid search in a 3D plot. Although

Setting	Value
Metadata element	"Title", "Director", "Role", "Persons", "ActorsAndRole", "Category", "Subtitle", "Synopsis", "Theme", "ChannelId", "Description", "Country"
Kerneltypes	Linear, Polynomial, Radial Basis Function
Termmode	Raw frequency, Term frequency, Term Frequency Inverse Document Frequency
RocSVM	true, false
Ignore case	true
Ignore stopwords	true
C	2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}
D	Standard value libsvm (default 3)
Gamma	Standard value libsvm (default $1/num\_features$)

Table 6.7: Test settings of the SVM approach.

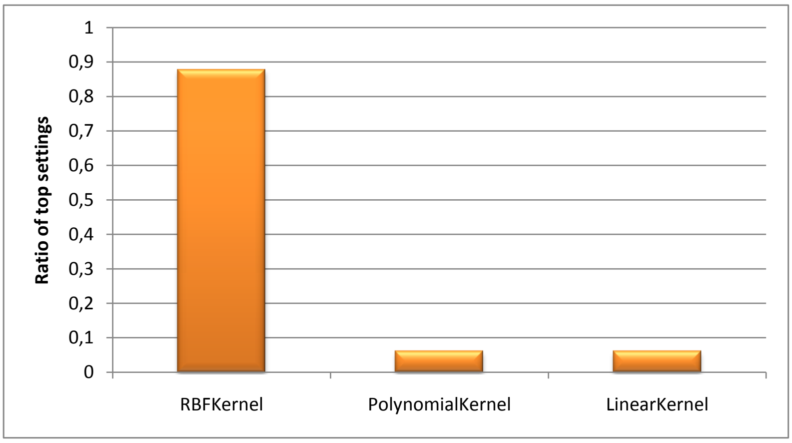


Figure 6.30: Overall ranking of SVM kerneltypes.

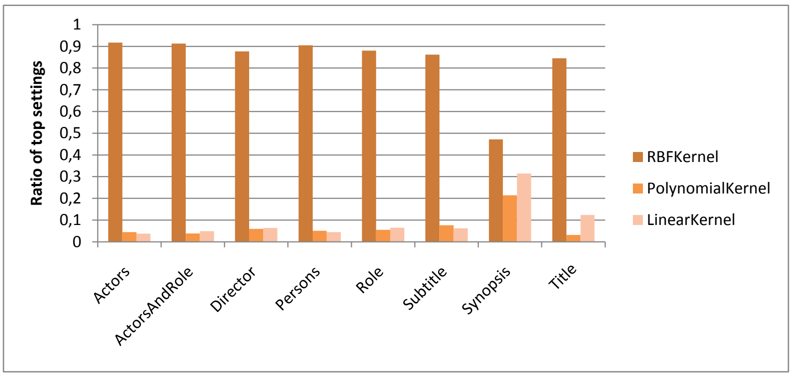


Figure 6.31: Ranking of SVM kerneltypes per metadata element.

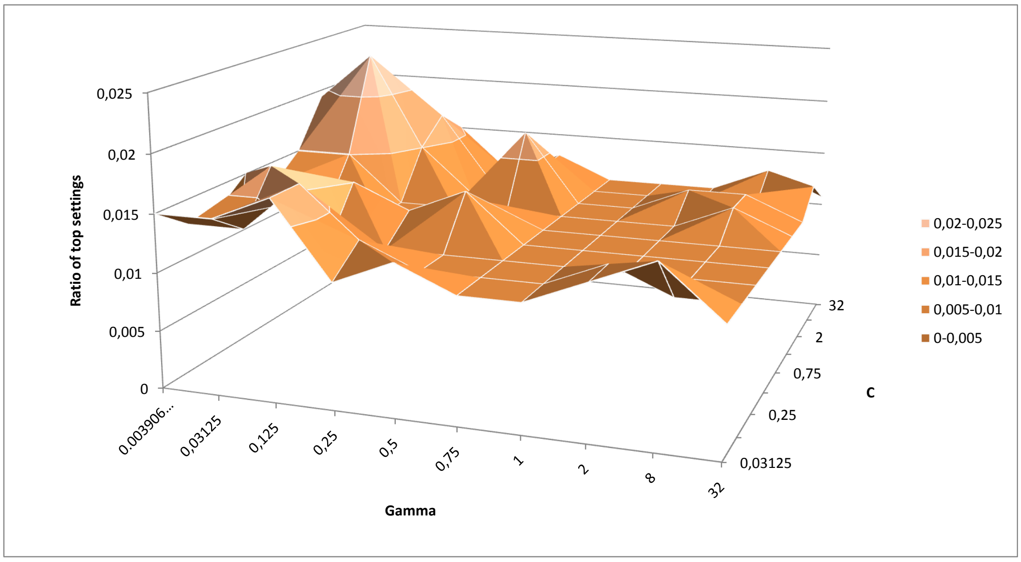


Figure 6.32: Gridsearch for parameter Gamma and C (RBF Kernel).

the differences between the settings are not very large, the best performance was achieved by $C = 2$ and $Gamma = 2^{-5}$, with a ratio of 2.5%. We were not able to determine any significant differences between the term modes. Thus the simplest method, the raw frequency, was selected. We suggest that the tf-idf mode should be favored in systems where no preprocessing steps, such as the elimination of common stop words, is conducted.

Having considered the use of the RocSVM approach for extracting ”reliable” negative sample, a surprising finding has been made: On average, the RocSVM approach slightly downgrades the classification results. Although it is clear that our assumption that a concurrent program is a negative sample (”uninteresting for the user”) is often not true, this simplification works quite well. In our user case study, it works even better than the RocSVM approach.

As a conclusion of this paragraph we recommend the use of the RBF Kernel with the raw frequency term mode and the value 2 for C and 2^{-5} for Gamma.

Latent Semantic Indexing Approaches

This paragraph evaluated the different settings of our LSI classification approach. A detailed description of this approach can be found in section 6.1.4 and 6.3.6. It should be mentioned that we decided to construct our LSI model (cf. "LSI Based Classifier" paragraph of section 6.3.6) based solely on the user's viewing history due to the results of the evaluation presented in table 6.6 and figure 6.28.

Figure 6.33 shows the XML Schema of the LSI classifier configuration. The schema allows for the setting of the following attributes:

- **knnNumber:** This setting defines the number of nearest neighbors considered in the calculation of the classification score. The scores of the nearest neighbors are combined according to equation (6.50).

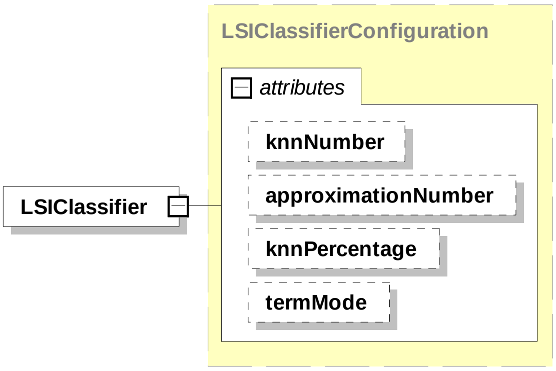


Figure 6.33: The LSI based configuration XML Schema.

- **approximationNumber:** The "approximationNumber" is used to specify the rank for the low rank approximation in the LSI process (cf. section 6.1.4).
- **knnPercentage:** The "knnPercentage" offers an alternative method to the "knnNumber" for setting the number of nearest neighbors as a ratio of the viewing history size.
- **termMode:** Similar to its counterpart in the SVM configuration, this setting determines the method for extracting and representing terms. Possible settings are RF, TF and TF-IDF.

In order to compare different classifier settings, a valuation of more than 150 different classifier ensemble configurations has been conducted. To construct these configurations all valid combinations of the settings shown in table 6.8 have been tested.

As with the term modes in the SVM configurations, no significant differences in the ratio of top LSI settings with different term modes could be found. Thus, raw frequency has also been chosen here Figure 6.34 shows the results of our sparse grid search conducted on the LSI settings KNN and approximation size. According to the discussion in [Ros00], we experienced very good results with low approximation sizes. The best ratio (19.21%) was achieved at a KNN value of 1 and a approximation size of 10. This configuration was chosen as the best overall LSI classifier ensemble configuration.

Setting	Value
Metadata element	"Synopsis"
Termmode	Raw frequency, Term frequency, Term Frequency Inverse Document Frequency
Ignore case	true
Ignore stopwords	true
KNN	$1, 2, 3, 4, 5$
Approximation size	$10, 50, 80, 90, 100, 150, 200, 250, 300, 400$

Table 6.8: Test settings of the LSI based approach.

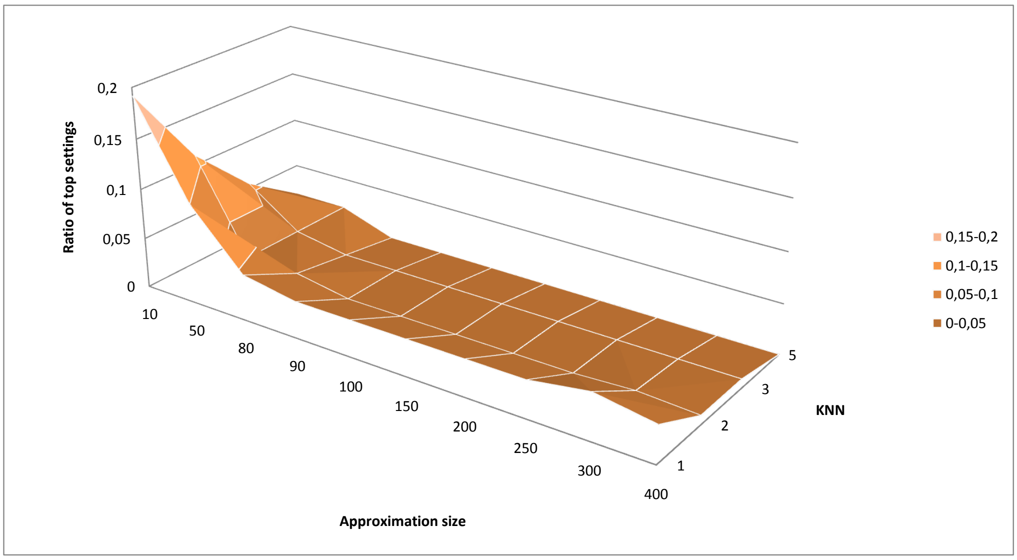


Figure 6.34: Gridsearch for parameter KNN and approximation size.

Comparison of Classification Approaches

The classification precision and the time needed for training and generating recommendations are two important factors in every recommendation system. This paragraph examines both factors and presents a comparison between the best configurations of the classification approaches presented in this work.

First we detail the runtime measurements. All tests were conducted on a single computation node of our high performance cluster in a single threaded manner. Each classifier approach was evaluated in a 4-fold-cross-validation on one profile with 372 programs and 10 runs each. Table 6.9 shows the average time needed for each main step of the classification approaches. Our spam based classifier provides the best runtime performance in all stages of the classification process. It is almost eight times faster than the SVM and thirty two times faster than the LSI based approach considering the overall runtime per cross-validation run. Please note that the overall runtime also includes the overheads for object

	Spam Filtering Approach	Support Vector Machine Approach	Latent Semantic Indexing Approach
Avg Building index / program	–	–	137.9 ms
Avg Training time / program	1.1 ms	8.2 ms	104.9 ms
Avg Classification time / program	1.0 ms	9.3 ms	66.3 ms
Overall Runtime (Cross-Validation Run)	55 s	424 s	1750 s

Table 6.9: Runtime of our classification approaches.

initialization, tokenization, pre- and post-processing. Compared to the other approaches, the LSI classifier is listed with an additional step for building the term-document index. Comparable steps can also be found in both of the other classification approaches. However measured per program the time needed in these approaches is negligible. Although the runtime of the approaches differ heavily, the times needed for processing one program indicate that all approaches could be successfully applied in a TV recommendation system.

Classifier type	Metadata element	AVG classification score	AVG position	Total positions in scope
RF Classifier	Category / Genre	0.855	13.68	63
RF Classifier	Title	0.760	16.52	63
RF Classifier	Synopsis	0.751	20.76	63
RBF SVM	Synopsis	0.192	21.44	63
RBF SVM	Title	0.424	22.67	63
RBF SVM	Category / Genre	0.235	25.64	63
LSI	Synopsis	0.719	27	63
RF Classifier	Persons	0.618	34.13	63
RF Classifier	Subtitle	0.542	37.16	63
RF Classifier	Actors	0.603	37.87	63

Table 6.10: Top 10 classifier settings and results on a single metadata element.

Table 6.10 shows the ten best classification results achieved with ensemble configurations using a single metadata element. It lists the classifier type, the metadata element used in the classification run and the average score achieved in the cross-validation test. The average position describes the average rank of each program (from the viewing history) among the average total number of concurrent program (”Total positions in scope”). Please note that all channels included in the intersection of all viewing histories have been used for the selection of concurrent programs. Looking at this table, it is interesting to note that our Spam based classification approach yields the best results although it uses the least complex classification model and performs best in terms of its runtime. The SVM as well as the LSI based approaches are not able to compete with the Spam based approach and yield lower scores for all metadata elements.

Please note that metadata element combinations yield much better results. For instance, if a SVM classifier (RBF SVM) is used on a combination of the top five metadata elements it achieves an average position of 11.78 compared to 8.7 of the RF Classifier. Thus, the ranking of classifier types (as shown in table 6.10) even holds for metadata element combinations.

6.5.4 Course of Classification Results and Related Implications

In order to quantify the adaptation power of the system with respect to the user’s preferences, we have considered the following indicators:

- Which metadata element is most relevant for the classification of the average user?
- One of the main issues of implicit learning algorithms is the duration of the training

phase that the algorithm needs before it is able to yield sensible results - the cold start phase. How many learned program descriptions are necessary to obtain the first promising recommendations?

- The course of the viewing history's average score and position: it is expected to increase with each additional piece of information learned, except for changes of the viewing habits. This score is further averaged using all selected viewing histories. It is expected that results will stay within the range of 0.7 to 1.0. Nevertheless, as several interesting programs may be shown at the same time, the viewer's choice may also have a slight negative impact on the scores of the other recommendations.
- Is the system able to adapt to user profiles in cases where viewing habits change?
- How does a context-sensitive setting change the course of the viewing history's average score and position?

For the purpose of evaluating the adaptation power, we used 40 watching histories with at least 80 programs each. The learning and classification of programs was done in an alternating manner. This means that the viewing histories were learned iteratively, by increasing the number of learned programs in each step by one. After each learning step, all 80 programs were classified and the average of the classification score and the average position was calculated. It should be noted that with this test setting the test and the training parts were not separated.

Figure 6.35 shows the course of the viewing history's average score and figure 6.36 the course of the average positions for each of the metadata elements ”ActorsAndRole”, ”Category/Genre”,”Country”,”Director”,”Persons”,”Role”,”Subtitle”,”Synopsis” and ”Title”. As a classifier, the RF classifier with the Beta confidence method was used, along with the ”Single_count_with_bonus_for_multiplies” token weighting method, smoothing of hapaxes and the window token selection method with a window size of 40. In both figures the three metadata elements (”Title”, ”Synopsis” and ”Category/Genre”) clearly outperform all others, which can be seen as an indicator of their overall importance for the ”average” viewer. When considering the average position, the best metadata element is the title. After learning about half of the programs (43) of the viewing histories, it achieves an average position of 10 for the whole history.

To determine the duration of the cold start phase we combined the five best metadata elements in one classifier ensemble. Figure 6.37 shows the course of the average positions for the metadata element combination ”Category/Genre”,”Persons”,”Subtitle”,”Synopsis” and ”Title”.

By comparing figure 6.36 and figure 6.37 it becomes clear that the combination of the metadata elements in one ensemble yields significantly better average positions. After learning about 1/3 of the programs (27 programs) the average position for the classification of the whole programs drops below 10. In looking at the average classification scores, the same observations about position hold. It should be noted that all metadata elements in this test are equally weighted. The course of the average score of this setting is shown in figure 6.40 (green line - unweighted). After the training of only 18 programs the average score reaches 0.7, which can be seen as the minimum score for recommendations. Although our classification approach treats all concurrent programs (on channels included

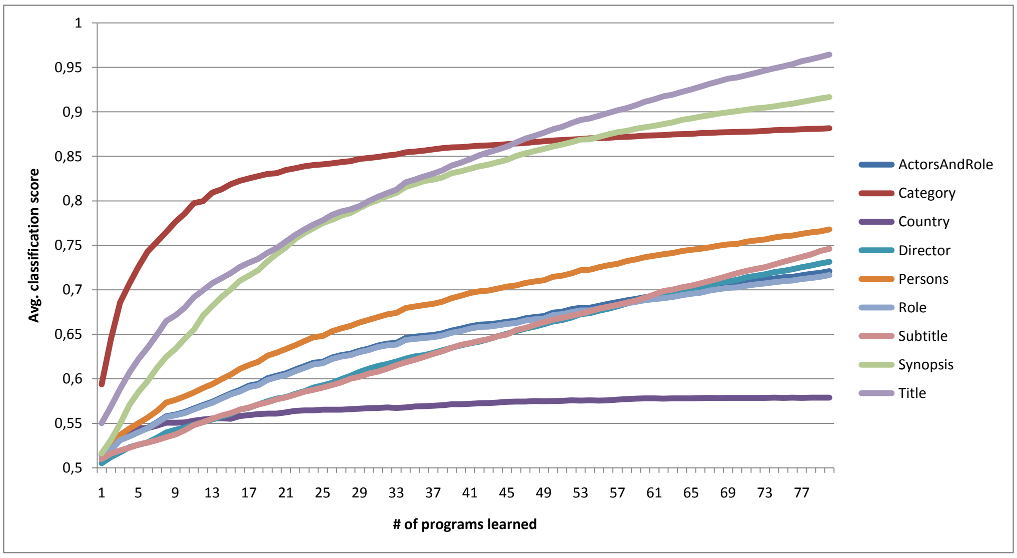


Figure 6.35: Trend of average classification scores.

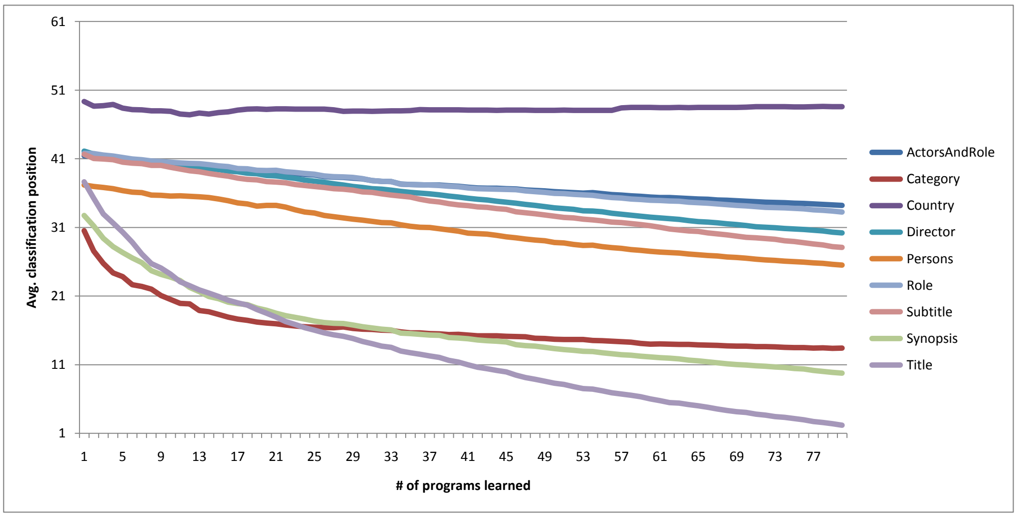


Figure 6.36: Trend of average position.

in the watch history) as negative training samples, the average score stays above 0.7 and finally reaches 0.86. At the end of the learning period an average position of 1.9 is reached. Moreover, the results of this evaluation and the good performance of this ensemble configuration have also been confirmed by a 4-fold-cross-validation with an average score of 0.74 and an average position of 8.7 among the average number of 63 concurrent programs.

Considering the ability of our system to adapt to changes, including varying user preferences or even a complete change of the user's preferences, we conducted a ”crossover” test of user profiles. First training and classification is done in the typical alternating manner. After 30 iterations the actual viewing history is substituted by the history of

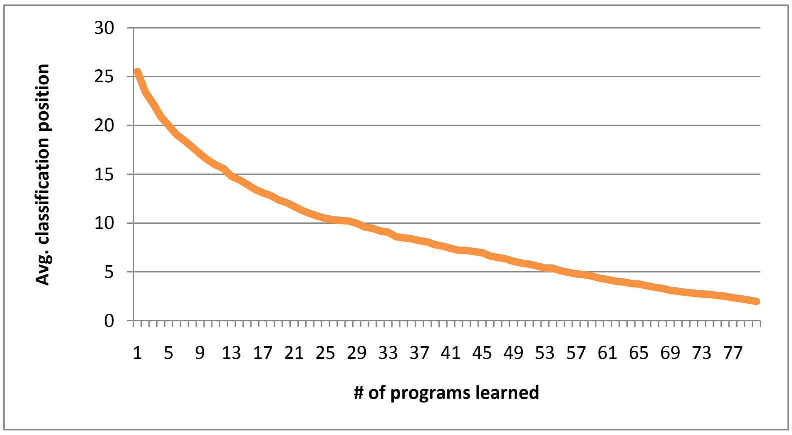


Figure 6.37: Combination of metadata elements (average position).

another user. We had expected that the classification performance would be affected by this change, but the system will adapt to the new user preferences very fast. Figure 6.38 shows the average scores and positions for two very dissimilar watch histories. Dissimilarity is measured by the outcome of a weight adaptation run, in which the weighting of the metadata elements of these histories were very different. As expected, the course of the average scores and positions strongly decline at 30 learned programs. After the training of about 5 additional programs of the second viewing history, our system starts to adapt to the new user history very fast. In general, decline at 30 is much smaller or sometimes not noticeable at all if the viewing histories are very similar.

To determine the impact of context-sensitive settings, we evaluated an exemplary situation, in which a differentiation between weekdays and weekends has been made. All tokens where annotated with a prefix ”weekday@” or ”weekend@”. For each context, a separate classification model was trained. A common model that does not differentiate between contexts has also been trained. We discovered that the best results can be achieved by using both, the context and the common model, by combining the token scores of both models. Using solely the context specific models can lead to a slightly poorer performance. We assume that this is a result of the fact that the results from the

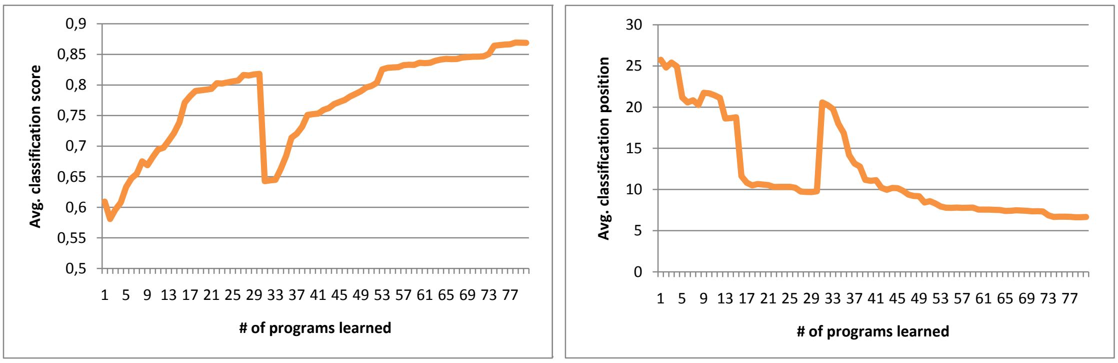


Figure 6.38: ”Crossover” test on two exemplary watch histories.

.

user’s behavior during the week affects those results gathered on the weekend, and vice versa. Thus, the use of a solely context specific model leads to the loss of such ”related” knowledge. Figure 6.39 shows the evaluation results. In the figure, the course of the context-sensitive setting’s score is always higher than those of the context-agnostic setting. As the amount of programs learned rose (> 60 programs), both methods perform equal. Nearly the same behavior can be observed in looking at the trends for the average position.

6.5.5 Dynamic Weighting of Classifiers

Classification based on single metadata elements allows for a fine tuned adaptation to the user’s individual interests. In the following section we examine the power of our dynamic weighting approaches. Again, we used 40 watching histories with at least 80 programs each in this test. For a detailed discussion of the proposed approaches, the reader is referred to section 6.3.6 paragraph ”Dynamic Weight Determination for Classifier Ensembles”. Figure 6.40 shows a comparison of a unweighted classifier ensembles with the top five metadata elements (cf. section 6.5.4) and the two most promising dynamic weighting approaches. The ”ModifiedWeight” approach of this figure complies with the approach described in equation (6.52) and the application of the *logit* function. The results of the biased feed-forward neuronal network are shown as ”NN.” Although the unweighted ensemble is configured with best performing metadata elements, both weight adaptation approaches are able to further enhance its performance. Moreover, the ”NN’ approach outperforms the ”ModifiedWeight” approach slightly. In looking at the average score, the ”NN” approach enhances the score of the unweighted ensemble by 7.8% and the ”ModifiedWeight” approach by 3%. With the rising number of different metadata elements combined in one classifier, the profit of the dynamic weight adaptation is even larger. For instance, in a test run with ensembles which incorporates 9 different metadata elements, the enhancement of the ”NN” approach showed a performance gain of more than 13%.

If the average course of weights (cf. figure 6.41) is closely examined, it can be seen that the individual metadata element weights are continuously adapted with each learning step. After about 30 programs the weights of ”Synopsis” and ”Category” are almost stable whereas the weight of ”Title” continues to rise until the end of the test. By modifying

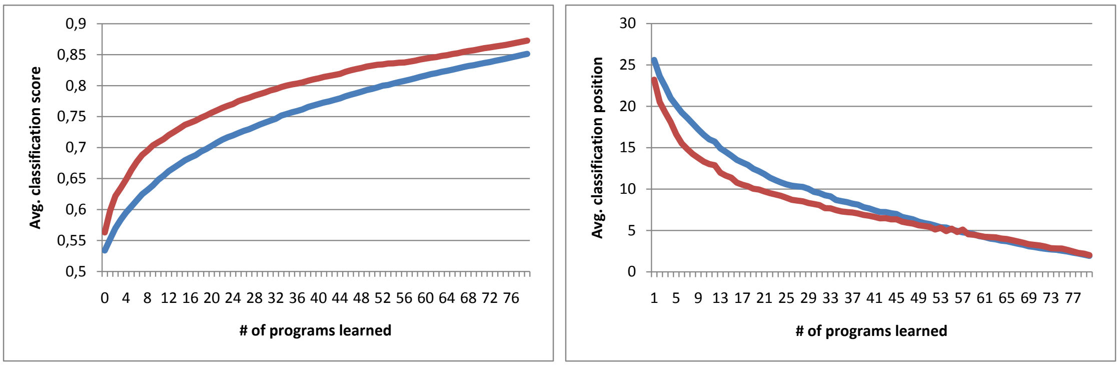


Figure 6.39: Context-sensitive setting (red) vs. normal setting (blue).

.

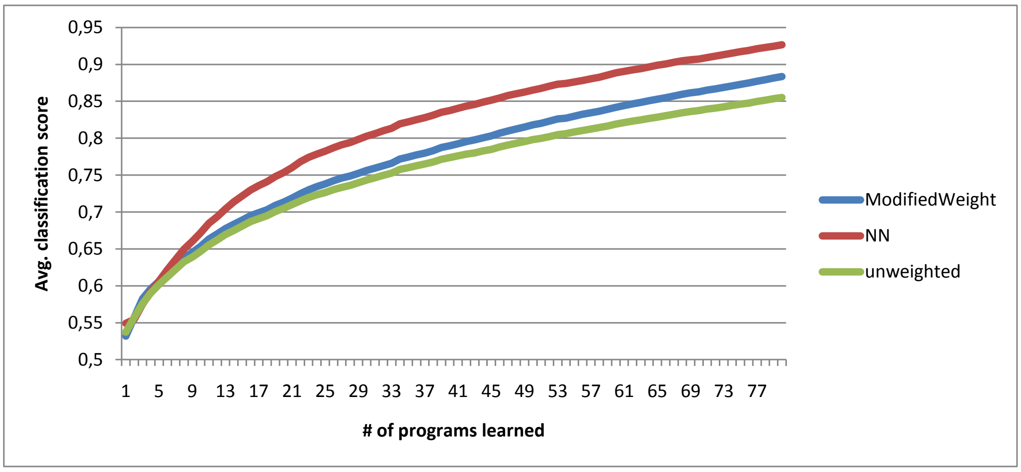


Figure 6.40: Dynamic weighting (average score).

the learning rate of the "NN," the course of the weight adaptation can be influenced. Generally, a high learning rate leads to faster adaptation, but introduces the risk of a higher weight variation during the adaptation process.

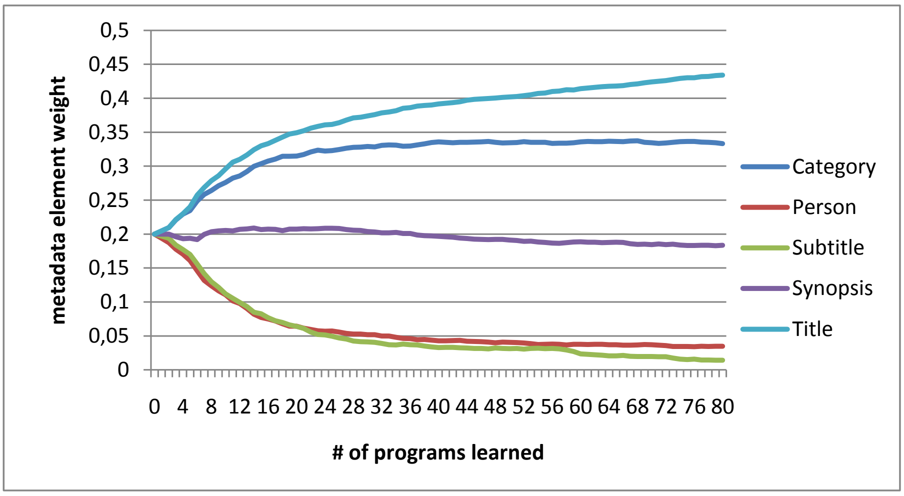


Figure 6.41: Course of weights for each metadata element (neuronal network approach).

As shown in figure 6.40, the dynamic weight adaptation improves the overall classification performance. Especially when user's viewing habits vary, this approach is highly appreciable. When examining the weights for individual users, a strong level of variation has been revealed which can be seen as an indicator for considerable differences in the viewing habits and the factors an individual considers to be decisive in choosing to watch a specific program.

6.5.6 Impact of the Advanced Tokenizer

The evaluation in this section examines the impact of our advanced tokenizer approach. A detailed description of this approach can be found in section 6.1.1. For the following evaluation the ”Synopsis” element of each program was processed by our tokenizer. A cross-validation test was conducted on the tokenizer’s result. Table 6.11 shows an excerpt of our test run. All results shown in this table should be examined in comparison to the results of the metadata element synopsis long (average score: 0.751, average position: 20.76) presented in section 6.5.3 paragraph ”Comparison of Classification Approaches”. The first entries such as ”Location only” where only the extracted locations and ”Token pairs” where only the extracted token pairs are used seem to be a bit disappointing. Nevertheless, even these settings might be valuable in heavily resource-constrained situations, because they require far fewer tokens. For instance, the ”synopsis” element has about 101.43 tokens on average per program, compared to 4.23 for the Named Entities (NEs), 12.04 for all token pairs and 1.53 for the locations.

Metadata element and setting	AVG classification score	AVG position	Total positions in scope
Locations only	0.572	37	63
Token pairs	0.568	25.72	63
NE’s only	0.648	22.84	63
Synopsis lemmatized	0.824	20.75	63
Synopsis lemmatized + NE’s	0.859	20.73	63

Table 6.11: Impact of the advanced tokenizer on the classification results.

The most surprising result is the good performance achieved in the classification of the Named Entities. Generally, there are very few NEs per program, compared to the number of tokens provided by the element ”synopsis.” Thus, NEs alone could be used in the classification process, instead of the element ”synopsis” without a considerable decrease in the performance of the classification. The use of lemmatized tokens from the ”synopsis,” as well as their combination with the NEs, provide only marginal improvements in the average classification position.

6.5.7 Evaluation Results - Collaborative Media Recommender

In order to judge the performance and to provide a means of comparison of our collaborative media recommender, we have conducted an experimental evaluation. All tests were performed on the dataset introduced in section 6.5.1. In the first paragraph of this section, some of the results from the TGE for upcoming programs will be presented. Then, current results based on the measures of precision and recall are given. All classifier ensemble settings used in this section are based on the evaluation results shown in sections 6.5.3 to 6.5.6. It should be noted that, due to runtime issues, advanced options such as dynamic weighting, context-sensitivity or the advanced tokenizer have not been used.

Sample results

An example of the tags generated by our TGE is plotted in table 6.12. Examining these tags in a qualitative manner, they seem to be very appropriate for the programs. Please note that the TGE generated different tag-clouds for two different episodes of ”The

Simpsons." In the first episode, the synopsis contains "George Washington," so the tag "amerika" was generated. The tags for "Anna und die Liebe" (a daily soap on German television) fit quite well too. "Galileo," which is a scientific news program, also generated the very appropriate tags "technik" (engl. technology) and "wissen" (engl. knowledge). Based upon these generated tag clouds and their overlaps, relationships between different programs can be uncovered and, as a result, adequate recommendations can be provided.

Tag	Simpsons 1	Simpsons 2	Anna und die Liebe	Galileo
amerika	0.820	-	-	-
cartoon	0.776	-	-	-
comedy	0.824	0.930	-	-
homer	0.831	0.920	-	-
humor	0.761	0.893	-	-
kult	0.821	0.956	-	-
lustig	0.802	0.852	-	-
serie	0.806	0.855	0.927	-
zeichentrick	0.838	0.954	-	-
liebe	-	-	0.999	-
spannend	-	-	0.996	-
technik	-	-	-	0.741
wissen	-	-	-	0.744

Table 6.12: Predicted tags and their scores.

Evaluation Results

In order to evaluate the tagging system, we tested the TGE on the client side (User Tag-Cloud) and on the server side (Global Tag-Cloud) separately. The tag generation on the client side was evaluated as follows: each user tag profile was split, according to its timeline, into a training set (75%) and a test set (25%).

To create a proper prediction setting, solely the most recent section of the history was used as the test set. To avoid randomness, a minimum number of programs was required for the training step. Because of this, only profiles with at least four tags that have been used for a minimum of five programs were selected. For analysis, we chose unranked precision-recall because we cannot provide a ranking on user assigned tags. Having performed this task, we reached a maximum f-measure of 0.36, although no pre-processing of the tags, such as spell checking or lemmatization has been done. Further tests with a typical splitting of 90%:10% resulted in an f-measure of up to 0.45.

The TGE on the server side was evaluated in a similar way. Here, however, we did not want to predict tags for a user profile, but tags which would be assigned to a program by all users. The dataset was split in a similar way. In this trail, we reached a maximum f-measure of 0.35.

Compared to the results of the AutoTag system [Mis06], our precision and recall values are a bit lower. Please note that for this, the scores of user and global TGEs have not yet been combined. Furthermore, we did not attempt syntactic matching among tags, as has

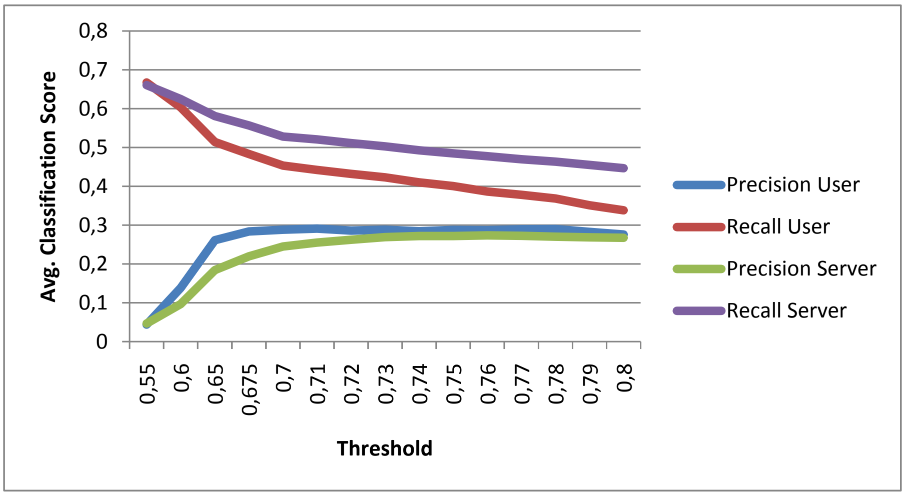


Figure 6.42: Threshold vs. precision and recall.

been done in the AutoTag system. We are also confident that restricting the number of tags to top-10 (as suggested in the AutoTag paper) would skew the precision values.

One of the major questions to be raised is: which threshold should be applied to the classifiers' scoring of a tag, before it can be added to the tag-cloud? For the answer, we refer to figure 6.42. The precision for both TGEs rises until a threshold of 0.71 (on the client side) and of 0.75 (on the server side) is reached. Then the recalls decrease whereas the precision remains constant. At a threshold of 0.77, the precision values on both sides start to decrease slowly. When looking at the maximum f-measures and the precision values, we suggest using a threshold of 0.71 on the client side and 0.75 on the server side. Please note that often, only the precision values are considered, as they are the most important factor in deriving good recommendations.

Another interesting question raised is: how does the number of tags used for classification influence the results? As there are sometimes too few tags in a user profile (in many cases less than 20) to create a response for the client TGE, we plotted only the TGE

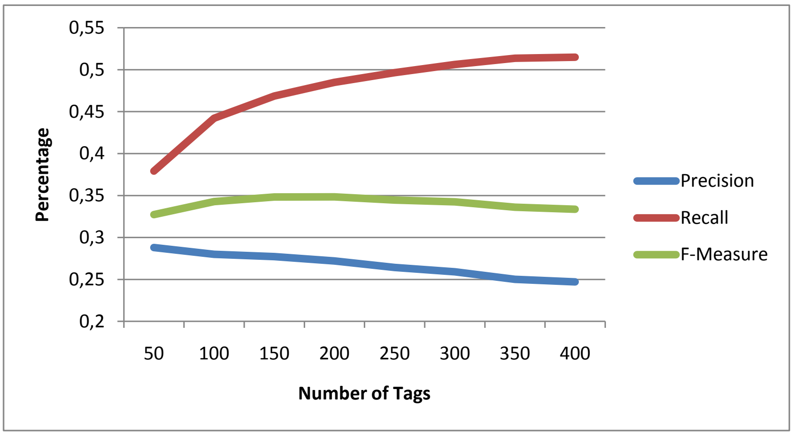


Figure 6.43: Number of tags vs. precision and recall on the server side.

on the server side. Nevertheless, the client TGE has also been considered in this test. Figure 6.43 shows that by increasing the number of tag-classifiers, the recall is improved significantly, whereas the precision is decreasing slightly. A good compromise between precision and recall can be found if the f-measure is taken into consideration. In figure 6.43 the maximum f-measure (0.349) is reached at a point when 200 tags are used in the classification process. On the client side, maximum f-measure of 0.351 is reached at 30 tags. Please note that when the number of tags and classifiers is increased, the amount of data that needs to be held in the memory of the computer increases as well. Thus, in roach we suggest using 200 tags on the server and 30 tags on the user side. For large scale application, we suggest dividing the dataset by tags and operating around 400 classifiers on a single server machine. Additionally, a reduction of the TGE's memory consumption can be accomplished using an aging approach, where very old program information is "unlearned" at a certain point. This approach also has an advantage as the meanings of certain words change over time. For instance, some, like "apple," may come to mean something different entirely.

In general, the number of available positive samples per tag should impact heavily the score values of the TGE. We have thus evaluated this impact using the ten most frequent tags, all of which have been used at least 45 times. In this experiment, approximately 1200 negative samples were used, compared to a maximum of 45 positive samples for a specific tag. Please note the devaluation of the negative examples described in section 6.4.4. Figure 6.44 shows the experimental results that demonstrate the correlation of the number of learned positive samples and the average TGE-score of the test set. On average, only about 15 positive examples per tag are necessary for a proper and accurate tag prediction with respect to the previously determined thresholds (cf. figure 6.42). In our experimental setting, where tag selection was not restricted and the number of users is small, it is quite hard to get enough (> 15) positive samples for the proper training of each tag TGE. Nevertheless, this problem can be faced by restricting the tag selection to a certain vocabulary. For larger systems with many contributing users, this problem is only noticeable in the long tail of the tag distribution (cf. figure 6.18).

To conclude this section, figure 6.45 presents our overall precision@10, recall@10 and

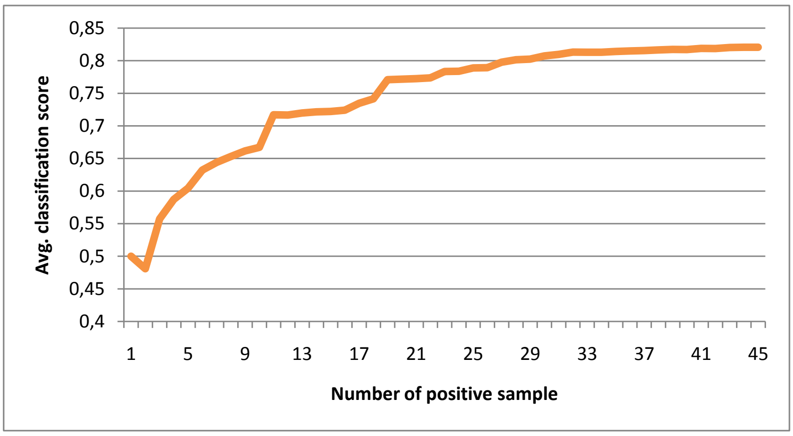


Figure 6.44: Progress of the average TGE-score with increasing number of positive samples.

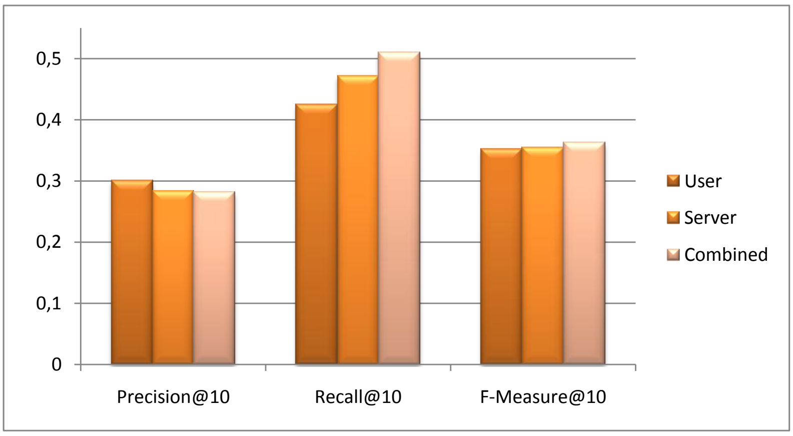


Figure 6.45: Precision, recall and f-measure @ 10.

f-measure@10 scores measured in a 4-fold-cross-validation. For the server TGE, we used 200 tags and a threshold of 0.75. 30 tags and a threshold of 0.71 were used for the user TGE. The combined values were calculated by adding the calculated user and server tag clouds in a 1:1 ratio. It is clear to see that the user TGE performs best with a value of 0.3 compared to 0.284 of the server TGE and 0.282 as a combined value. Looking at the f-measures, all three modes used featured very similar performances. Larger differences can be seen in recall@10. Here, the combined mode, with an value of 0.51, outperforms the server (0.472) and the user TGE (0.425). This result mainly stems from the fact that in the user TGE, only a few tags are considered, compared to the combined mode where both the user and the server tags are used.

Furthermore, the impact of enhanced tag preparation (cf. section 6.4.5) using GermaNet and Leacock-Chodrow Measure was also evaluated. Unfortunately the tag preparation was not able to noticeably enhance our results. Although relationships, such as ”humor” is closely related to ”fun,” ”agent” to ”secret agent,” and ”war” to ”fight” were uncovered, the total number of these relationships was far too small. The main reason behind this was that the matching of only a few tags and synsets was possible, because of the use of many colloquial and compound words. Additionally, the appearance of many foreign language tags further complicated the matching process. Nevertheless, also without using this enhancement our recommender achieves a good performance as compared to approaches like AutoTag [Mis06].

CHAPTER 7

Conclusion and Future Prospects

Multimedia consumption and systems form a widespread area for research and a field with many potential new innovative applications. With the ever increasing use of mobile devices and the tremendous growth in amount and variety of available content, new concepts for multimedia consumption need to be found. In this work, we presented the concepts of user-centrism and applied them to the field of multimedia systems. Furthermore, we developed our own definition of a user-centric multimedia framework. To evaluate the applicability of these concepts, an evaluation platform which allows for personalized means of media consumption has been implemented. It is comprised of new concepts for supporting session mobility in a seamless way, additional personalized services and a novel selection support of multimedia content. We were able to demonstrate the feasibility of our key concepts, such as our novel recommendation component with its dynamic adaptation and confidence approaches and our metadata enhancement, by focusing specifically on the realm of TV and of multimedia streaming.

To allow for the transfer of multimedia sessions, a novel component has been developed which is comprised of a mobile agent system and parts of the standardized description of the MPEG-21 Multimedia Framework initiative. MPEG-21 has been used to describe and encapsulate the session data and its context. As proof of this concept, a use case for video session migration has been implemented. Our session mobility component can, more generally, be used with various session types, such as ”browsing sessions”. For each session type an individual strategy for the selection of a migration target among all available device for continuing the session can be realized. For instance, for the video session type a strategy has been implemented which takes the media capabilities (e.g. available codecs, screen resolution and color depth) and the CPU speed (measured by a simple benchmark) of the target device into consideration. For our user-centric multimedia framework this component enables mobility of the personalization component and additional services.

Interactive TV add-on services are provided by our iTV component in an original way. These services can be used on a personal mobile device such as a smartphone, tablet or handheld in a synchronized (e.g. participate in a game show or votes) or asynchronous manner (gathering additional information on news topics). To guarantee easy and intuitive access to our services, the Universal Plug and Play (UPnP) specification has been used. Thus, services are automatically discovered and easily controlled. Moreover, an almost configuration free access is possible. The capabilities of this component have been demonstrated with two use cases, the ”game show scenario” and the ”news ticker scenario”. For our user-centric multimedia framework, this component enables an easy access to different services (e.g. a personalization service) and the use of add-on services for multimedia streaming and broadcast.

One very important issue in multimedia consumption is the discovery and selection of interesting and appropriate content. With personalTV, a component for personalized selection support has been developed. This is a novel way of combining different classification (Support Vector Machines), filtering (Spam Filter) and text-mining (Latent Semantic Indexing) mechanisms with tagging (a typical Web 2.0 mechanism). To provide accurate recommendations, all mechanism have been transfered and adequately adapted to our media recommendation approach. Additionally, a concept for dynamic adaptation of personalTV in accordance with changes in the current situation (available metadata, device, time, etc.) has been proposed. To enhance recommendation quality, an approach has been presented for processing and analyzing textual inputs that facilitates the identification and extraction of important semantic concepts. In order to accomplish this task, several techniques from the fields of Text Mining and Natural Language Processing (NLP) have been combined. A basic confidence measure of program predictions has also been introduced to account for uncertainty in the classification process. Here, we have again used the realm of TV as our focus to demonstrate the applicability of our concepts. Personalized recommendations are generated based upon the analysis of an individual user’s history of watched and tagged TV programs and, optionally, his or her explicit profile. All concepts introduced have been extensively evaluated on our TV metadata test corpus which consist of 67 users’ viewing histories with a total of 10,845 programs, collected over a period of 10 month. This evaluation revealed interesting facts such as the outstanding results of our Spam filtering based recommendation approach.

In the following section the ”Future Work” paragraph outlines the most important extensions and enhancements of this work. As a conclusion, we present our vision for media consumption with a focus on the realm of TV.

Future Prospects

Based on the work presented in this thesis, several interesting extensions may be realized. In the following section, we survey the most important of these.

- **Extension of the implementation to media in general:** All concepts presented in this thesis have been evaluated in the realm of TV. Nevertheless, they could be easily used for multimedia content in general as they require only the availability of adequate metadata and a network connection. Using our components such as the recommendation system for media consumption beyond the context of television (e.g. when consuming other media content or surfing the internet) would allow for an extension of the usage histories. For instance, through the use of the session mobility component, different usage contexts with respect to e.g. the current location, device or media type could also be included in the recommendation and adaptation process of the system. Thus, the common interest of the user could be taken into account. Additionally, further user actions could be integrated and interpreted in our system. For instance, browsing the EPG, intense zapping, partially watched programs or even actions like surfing the web or buying a product could further extend the knowledge base for reasonable recommendations.

- **Integration into an iTV middleware platform:** Section 2.1.3 presented several of the most prominent, open iTV standards and middleware platform specifications. The level of development of most of these platforms, however, needs to be rated as mature. Nevertheless mobile devices and second screen mechanisms are addressed only marginally, if at all. Thus, the integration of our concepts which focus on mobile devices can be a valuable contribution to the success of these platforms. The integration of our recommendation system, which has been designed with resource-constrained environments in mind, could enhance usability considerably.

- **Relevance Feedback:** Users should be able to understand, at least to some extent, why a specific program is recommended by the system. That is, the recommendation system must be able to explain it to them. This has the potential to simultaneously enhance the system's popularity and to help improve the recommendation process. Making the evolution process of recommendations clear to the viewer is not that far removed from the relevance feedback approach. The weighting factors of the single evaluation components can be determined by asking the user to rank a set of favored programs and then further adapting them in an iterative process. Explicitly asking users for their preferences concerning selected programs might help to identify new likes and dislikes. At this point, we suggest that mechanisms from the area of active learning such as active sample selection strategies or outliers detection (cf. [Sou05]) should be integrated. For instance, we suggest that by applying an active sample selection strategy the number of samples needed to achieve a reasonable classification quality can be strongly reduced (cf. the reductions reported in [Aya07]). However, the success of such approaches is very dependent on the datasets, and as a result more detailed investigations are required. First, an initial user profile based upon the evaluation of the user history could be created. In considering this profile, program lists could be compiled and presented to the user. The initial profile would then be modified through a feature-based analysis according to the selected programs, so that the profile can be adjusted step by step to the user's true preferences. Including these user rated programs in the implicit profile seems to be a promising technique to shorten the cold start phase and to increase the system's recommendation quality.

- **Taxonomy generation and the application of bigger datasets:** Based on big sets of tags and tag clouds, analysis used in data mining should be capable of discovering a broad variety of relationships between different tags. These relationships can be incorporated into the recommendation process to produce more accurate and less ”over-specialized” recommendations. ”Over-specialized” means that the approaches tend to solely recommend items the user has liked in the past or that are at least very similar to such items. This leads to a very limited variation within the recommendations. Furthermore, our evaluation results clearly indicate that with more training data, the relation between programs, tags and metadata could be more precisely uncovered. Thus, the utilization of bigger datasets such as the MovieLens[1]

[1] MovieLens - http://www.movielens.org/

dataset augmented with movie metadata available from movie databases such as the IMDb might provide valuable results.

Vision

The recent history of multimedia, and particularly the realm of TV, has brought many technological changes. After more than 30 years of different trials and attempts to establish HD TV, it seems finally to have become part of mainstream technology. 3D-TV is also back once again. These developments are discussed widely in the media branch and plain for anyone to see. Please note that the Video and DVD branch is not the focus of the following discussion.

Nevertheless the main ”media revolution” is happening on a more subtle level. From a historic point of view, the world of media has been clearly separated into the TV or broadcast domain, and into the Web or streaming domain. Recently, this distinction becomes more and more blurred.

On the one side, web platforms and services such as YouTube[1] and MyVideo[2] offer alternative content to the TV program and are increasingly successful in rivaling traditional TV. Furthermore, offers such as Zatto[3], TVCatchup[4] or Hulu[5] compete with TV by using its ”own” content by providing series, movies or even entire TV channels to an audience via the web. The field of movie rental is also becoming more and more competitive because of online movie rental platforms like Netflix[6].

On the other hand, the TV branch has also noticed these developments and tries to use the internet and its resources as a kind of add-on offer. Broadcasters recently developed the so-called ”Catch-up-TV” services where users can access the programs from the last few days on web portals. Examples for such portals are the ZDF Mediathek[7], ARD Mediathek[8] or even mobile solutions such as the BBC iPlayer[9]. In contrast to offering TV content on the Web, hydride TV solutions such as HbbTV (cf. section 2.1.3) enhance the traditional TV program by integrating internet resources and services into the TV program and the TV-set.

Until this point, none of the mentioned development is an urgent risk to the main business model (advertisement) of the broadcasters. Nevertheless, developments of several manufacturers and their cooperation partners, such NetTV[10] or Yahoo Connected TV[11], are a huge risks for the business of most broadcasters. This is because with these technology the control of how, where and when an ad is shown is no longer at the broadcasters side. In such systems, every area of the screen can be used for other applications or even for

[1] YouTube – http://www.youtube.com/

[2] MyVideo – http://www.myvideo.com/

[3] Zatto – http://zattoo.com/

[4] TVCatchup – http://www.tvcatchup.com/

[5] Hulu - http://www.hulu.com/

[6] Netflix – http://www.netflix.com/

[7] ZDF Mediathek – http://www.zdf.de/ZDFmediathek/

[8] ARD Mediathek – http://www.ardmediathek.de/

[9] BBC iPlayer – http://www.bbc.co.uk/iplayer/

[10] NetTV – http://www.nettv.philips.com/

[11] Yahoo Connected TV – http://connectedtv.yahoo.com/

ads provided by them. Thus, the broadcasters are not able to guarantee that the ad or logo of their advertising consumer is visible to the audience anymore.

This development seems to be further sped up by systems such as GoogleTV[1] and AppleTV[2] where the role of the traditional broadcaster is downgraded to that of a content provider, and may even be completely eliminated in future. With these systems, TV-related and other functionality can be easily extended with additional applications and services. Furthermore, the integration of mobile devices, such as different types of android phones, iPhones and iPads, is already an integral part of these systems. The concept of these systems seems to be particularly promising in terms of usability. Video game consoles, such as the Microsoft XBox 360 and the Sony Playstation 3, are new competitors in the TV and video market by integrating typical video and TV functions. In the future, these systems may even further replace traditional TV.

To me, it seems quite questionable that the recent efforts of the broadcasters, such as HbbTV which is restricted in aspects like the placement of the video and the integration of third-party web content, are really competitive to the well integration of web and TV by contenders such as GoogleTV.

Not only the business model of the broadcasters, but also the role of the audience in the value chain is changing. The Web 2.0 philosophy has already arrived in the media domain. YouTube is the best example of the fact that the consumer may also act as the producer - the so called ”produser” (or even ”prosumer”). Particularly in news programs, produser content (e.g. amateur videos of the tsunami in Thailand) is already used in the world of professionally produced media content. We suggest that in the next years the amount of produser content in the area of professional media will rise continuously. Although some sources already talk of the end of professionally produced video content, we do not think that professional productions will be substituted completely, rather enriched by produser content. In systems such as GoogleTV, the role of the produser is gaining further importance because of the direct integration and access to services and portals like YouTube and Flicker. Additionally, interactive videos and TV are still an important topic. As a result of the integration of Web features, a good foundation for offering such content is available. Here as well, more and more content is created by produsers (cf. YouTube Annotations[3]). Thus, the final establishment of interactive videos and interactive TV might come as a result of these new developments.

Although Ruhrmann et al. [Ruh97] coined the term ”Communicative TV,” the impact of social communities on TV in 1997 was only marginal. With the enormous success of different social networks such as Facebook, Myspace or Twitter, the integration of these communities into TV is also advancing quickly. First commercial products include social features. For instance the Microsoft XBox 360 allows users to watch TV together with their friends, each represented by an avatar in a virtual living room. Here as well, the sharing of experiences is an important concept. Even in the research community, social media and social TV is one important topic and the focus of many publications (cf. ”Social Media Tracks” on the euroITV conferences). Most TV middleware systems plan to or

[1] GoogleTV – http://www.google.com/tv/

[2] AppleTV – http://www.apple.com/appletv/

[3] YouTube Annotations – http://www.youtube.com/t/annotations_about

already have integrated social functions like chats, video telephony, social network services (e.g. facebook) or even provide recommendations of the users social network.

Looking at these developments and at the future prospects of the media branch, it seems that the way to consume media content will soon be very different, when compared to traditional TV. The integration of social communities will play an important role in the realm of media consumption. Moreover, the typical broadcast distribution channel, such as terrestrial and cable, will also lose importance because of the strong rise of high bandwidth internet connections. Approaches such as Fiber-to-the-home (FTTH) where households are directly connected with a fiber to the internet featuring bandwidths of up to 1 Gbit/s can easily replace most other distribution channels in an integrated manner. Furthermore, we suppose that interactivity in TV and videos now has the potential to become mainstream, even in countries where it has hardly been present up to now. The fusion of internet and TV content will make it easier to provide interactive service and as a result, pave the way to a broad offering of such content and services. Nevertheless, we think that the ”lean-back” experience of TV and media consumption will still be dominant. However, the lingering main question behind the continuously expanding plethora of content seems to be: how can feasible ways of finding content of interest be provided to users? In our opinion, the only way to face this challenge is to develop and apply hybrid media recommendation systems. Moreover, we are quite sure that the time of ”traditional” TV schedules created by an editorial staff is quickly coming to an end. We believe that future TV schedules will be a mixture of programs recommended in a social manner (by the community of the user also called ”social recommenders”) and by recommendation systems trained upon the user’s preferences and behavior.

Bibliography

[Abb09] Abbassi, Zeinab; Amer-Yahia, Sihem; Lakshmanan, Laks V.S.; Vassilvitskii, Sergei and Yu, Cong: Getting recommender systems to think outside the box, in: *RecSys '09: Proceedings of the 3rd ACM Conference on Recommender systems*, ACM, New York, NY, USA, pp. 285–288

[Ade10] Adeyeye, Michael and Ventura, Neco: A SIP-based web client for HTTP session mobility and multimedia services. *Computer Communications* (2010), vol. 33:pp. 954–964

[Ado05] Adomavicius, Gediminas and Tuzhilin, Alexander: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* (2005), vol. 17(6):pp. 734–749

[Adv03] Advanced Television Systems Committee: DTV Application Software Environment Level 1 (DASE-1) Part 1: Introduction, Architecture, And Common Facilities (2003), (ATSC - A/100)

[Adv09] Advanced Television Systems Committee: Advanced Common Application Platform (ACAP) (2009), (ATSC - A/101A)

[Agi06] Agirre, Eneko and Edmonds, Philip (Editors): *Word Sense Disambiguation: Algorithms and Applications*, vol. 33 of *Text, Speech and Language Technology*, Springer (2006)

[Akk06] Akkermans, Paul; Aroyo, Lora and Bellekens, Pieter: iFanzy: Personalised Filtering Using Semantically Enriched TV-Anytime Content, in: *ESWC '06: Proceedings of the 3rd European Semantic Web Conference*, ACM, New York, NY, USA

[Asa03] Asahara, Masayuki and Matsumoto, Yuji: Japanese Named Entity extraction with redundant morphological analysis, in: *HTL-NAACL '03: Proceedings of the main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 8–15

[Aya07] Ayache, Stéphane and Quénot, Georges: Evaluation of active learning strategies for video indexing. *Image Communication* (2007), vol. 22:pp. 692–704

[Bag07] Bagci, Faruk; Schick, Holger; Petzold, Jan; Trumler, Wolfgang and Ungerer, Theo: The reflective mobile agent paradigm implemented in a smart office environment. *Personal and Ubiquitous Computing* (2007), vol. 11:pp. 11–19, 10.1007/s00779-005-0059-y

[Bal97] Balabanovic, Marko and Shoham, Yoav: Fab: content-based, collaborative recommendation. *Communications of the ACM* (1997), vol. 40:pp. 66–72

[Ban02] Banerjee, Satanjeev and Pedersen, Ted: An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet, in: *CICLing '03: Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing*, Springer-Verlag, London, UK, pp. 136–145

[Ban06] Banerjee, Nilanjan; Acharya, Arup and Das, Sajal K.: Seamless SIP-based mobility for multimedia applications. *IEEE Network Magazine* (March-April 2006), vol. 20(2):pp. 6–13

[Bär08] Bär, Arian; Berger, Andreas; Egger, Sebastian and Schatz, Raimund: A Lightweight Mobile TV Recommender, in: *EUROITV '08: Proceedings of the 6th European Conference on Changing Television Environments*, EUROITV '08, Springer-Verlag, Berlin, Heidelberg, pp. 143–147

[Bel07] Bellifemine, Fabio; Caire, Giovanni and Greenwood, Dominic: *Developing multi-agent systems with JADE*, John Wiley & Sons (2007)

[Ber95] Berry, Michael W.; Dumais, Susan T. and O'Brien, Gavin W.: Using linear algebra for intelligent information retrieval. *SIAM Review* (1995), vol. 37:pp. 573–595

[Ber08] Bernhaupt, Regina; Wilfinger, David; Weiss, Astrid and Tscheligi, Manfred: An Ethnographic Study on Recommendations in the Living Room: Implications for the Design of iTV Recommender Systems, in: *EUROITV '08: Proceedings of the 6th European Conference on Changing Television Environments*, EUROITV '08, Springer-Verlag, Berlin, Heidelberg, pp. 92–101

[BF07] Blanco-Fernández, Yolanda; Arias, José J. Pazos; Gil-Solla, Alberto; Cabrer, Manuel Ramos; Nores, Martín López; Duque, Jorge García; Vilas, Ana Fernández; Redondo, Rebeca P. Díaz and noz, Jesús Bermejo Mu Avatar: Enhancing the Personalized Television by Semantic Inference. *International Journal of Pattern Recognition and Artificial Intelligence* (2007), vol. 21(2):pp. 397–421

[Bik97] Bikel, Daniel M.; Miller, Scott; Schwartz, Richard and Weischedel, Ralph: Nymble: a high-performance learning name-finder, in: *ANLP '97: Proceedings of the 5th Conference on Applied Natural Language Processing*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 194–201

[Bje10] Bjelica, M.: Towards TV recommender system: experiments with user modeling. *Consumer Electronics, IEEE Transactions on* (2010), vol. 56(3):pp. 1763 –1769

[Bla98] Black, William; Rinaldi, Fabio and Mowatt, David: Facile: Description Of The Ne System Used For MUC-7, in: *MUC-7 '98: Proceedings of the 7th Message Understanding Conference*

[Bor98] Borthwick, Andrew; Sterling, John; Agichtein, Eugene and Grishman, Ralph: NYU: Description of the MENE Named Entity System as Used in MUC-7, in: *MUC-7 '98: Proceedings of the 7th Message Understanding Conference*

[Bra04] Braschler, Martin and Ripplinger, Bärbel: How Effective is Stemming and Decompounding for German Text Retrieval? *Information Retrieval* (2004), vol. 7:pp. 291–316

[Bri03] British Broadcasting Corporation: Digital Terrestrial Television MHEG-5 Specification (2003)

[Buc94] Buckley, Chris; Salton, Gerard and Allan, James: The effect of adding relevance information in a relevance feedback environment, in: *SIGIR '94: Proceedings of the 17th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, Springer-Verlag New York, Inc., New York, NY, USA, pp. 292–300

[Bud06] Budanitsky, Alexander and Hirst, Graeme: Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics* (2006), vol. 32:pp. 13–47

[Bur98] Burges, Christopher J. C.: A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery* (1998), vol. 2:pp. 121–167

[Bur02] Burke, Robin D.: Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* (2002), vol. 12:pp. 331–370

[Bur05] Burnett, I.S.; Davis, S.J. and Drury, G.M.: MPEG-21 digital item declaration and Identification-principles and compression. *IEEE Transactions on Multimedia* (2005), vol. 7(3):pp. 400–407

[Bur06] Burnett, Ian S.; Pereira, Fernando; de Walle, Rik Van and Koenen, Rob: *The MPEG-21 Book*, John Wiley & Sons (2006)

[Bur07] Burke, Robin: *The adaptive web*, Springer-Verlag, Berlin, Heidelberg (2007)

[Cab05] Cable Television Laboratories: OpenCable Application Platform Specification - OCAP 1.0 Profile (2005)

[Cal00] Calder, Bart; Courtney, Jon; Foote, Bill; Kyrnitszke, Linda; Rivas, David; Saito, Chihiro; VanLoo, James and Ye, Tao: Java TV API Technical Overview: The Java TV API Whitepaper, Technical Report, Sun Microsystems, Inc. (2000)

[CH08] Cleland-Huang, Jane and Mobasher, Bamshad: Using data mining and recommender systems to scale up the requirements process, in: *ULSSIS '08:*

Proceedings of the 2nd International Workshop on Ultra-Large-Scale Software-Intensive Systems, ACM, New York, NY, USA, pp. 3–6

[Chi03] Chieu, Hai Leong and Ng, Hwee Tou: Named entity recognition with a maximum entropy approach, in: *HLT-NAACL '03: Proceedings of the main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 160–163

[Cho06] Choi, Yoo-Joo; Yoo, Seong; Choi, Soo-Mi; Waldeck, Carsten and Balfanz, Dirk: User-Centric Multimedia Information Visualization for Mobile Devices in the Ubiquitous Environment, in: Bogdan Gabrys; Robert Howlett and Lakhmi Jain (Editors) *KES '06: Proceedings of the 10th International Conference on Knowledge-Based & Intelligent Information and Engineering Systems*, vol. 4251 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg (2006), pp. 753–762

[Con07] Consumer Electronics Association: CEA-2014 - Web-based Protocol and Framework for Remote User Interface on UPnP Networks and the Internet (Web4CE) (2007)

[Cor95] Cortes, Corinna and Vapnik, Vladimir: Support-Vector Networks. *Machine Learning* (1995), vol. 20:pp. 273–297

[Cor00] Corporation, British Broadcasting: Standard Media Exchange Framework (SMEF) Data Model 1.5 (2000)

[Cos03] Cosley, Dan; Lam, Shyong K.; Albert, Istvan; Konstan, Joseph A. and Riedl, John: Is seeing believing?: how recommender system interfaces affect users' opinions, in: *CHI '03: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, ACM, New York, NY, USA, pp. 585–592

[Cre10] Cremonesi, Paolo and Turrin, Roberto: Time-evolution of IPTV recommender systems, in: *EuroITV '10: Proceedings of the 8th International Interactive Conference on Interactive TV&Video*, EuroITV '10, ACM, New York, NY, USA, pp. 105–114

[Cye04] Cyeon, H.L.; Schmidt, T.C.; Wahlisch, M.; Palkow, M. and Regensburg, H.: A distributed multimedia communication system and its applications to E-learning, in: *ISCE '04: IEEE International Symposium on Consumer Electronics*, pp. 425–429

[Dau05] Daumé, Hal, III and Marcu, Daniel: A large-scale exploration of effective global features for a joint entity detection and tracking model, in: *HLT '05: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 97–104

[Dee90] Deerwester, Scott; Dumais, Susan T.; Furnas, George W.; Landauer, Thomas K. and Harshman, Richard: Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science* (1990), vol. 41:pp. 391–407

[Dey01] Dey, Anind K.: Understanding and Using Context. *Personal Ubiquitous Computing* (2001), vol. 5:pp. 4–7

[Dig94] Digital Audio-Visual Council: Statutes of The Digital Audio-Visual Council (1994)

[Din08] Ding, Wang; Yu, Songnian; Yu, Shanqing; Wei, Wei and Wang, Qianfeng: LRLW-LSI: an improved latent semantic indexing (LSI) text classifier, in: *RSKT '08: Proceedings of the 3rd International Conference on Rough sets and knowledge technology*, RSKT'08, Springer-Verlag, Berlin, Heidelberg, pp. 483–490

[Dum95] Dumais, Susan T.: Latent Semantic Indexing (LSI): TREC-3 Report, in: *TREC-3 '95: Proceedings of the Third Text REtrieval Conference*, pp. 219–230

[Eur94] European Telecommunications Standards Institute: Terminal Equipment (TE); MHEG script interchange representation (MHEG-SIR) (1994), (ETSI TS 300 715)

[Eur05a] European Broadcasting Union: EBU Core Metadata Set (2005), (EBU –Tech 3293)

[Eur05b] European Telecommunications Standards Institute: Globally Executable MHP version 1.0.2 (GEM 1.0.2) (2005), (ETS TS 102 819)

[Eur06a] European Telecommunications Standards Institute: Broadcast and On-line Services: Search, select, and rightful use of content on personal storage systems ("TV-Anytime"); Part 3: Metadata; Sub-part 1: Phase 1 - Metadata schemas (2006), (ETSI TS 102 822-3-1)

[Eur06b] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Electronic Service Guide (ESG) (2006), (ETSI ES 102 471)

[Eur06c] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.0.3 (2006), (ETSI TS 201 812)

[Eur06d] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.1.1 (2006), (ETSI TS 102 812)

[Eur07a] European Broadcasting Union: P_META 2.0 Metadata Library (2007), (EBU – TECH 3295-v2)

[Eur07b] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Globally Executable MHP (GEM) Specification 1.2.2 (including IPTV) (2007), (ETSI TS 102 728)

[Eur07c] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2 (2007), (ETSI TS 102 817)

[Eur08] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB);Specification for Service Information (SI) in DVB systems (2008), (ETSI EN 300 468)

[Eur10a] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Signaling and carriage of interactive applications and services in Hybrid Broadcast/Broadband environments V1.1.1 (2010), (ETSI TS 102 809)

[Eur10b] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Globally Executable MHP (GEM) Specification 1.2.2 (including IPTV) (2010), (ETSI TS 102 728)

[Eur10c] European Telecommunications Standards Institute: Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2.2 (2010), (ETSI TS 102 727)

[Eur10d] European Telecommunications Standards Institute: Hybrid Broadcast Broadband TV (2010), (ETSI TS 102 796 V1.1.1)

[Fal05] Fallahkhair, Sanaz; Pemberton, Lyn and Griffiths, Richard: Dual Device User Interface Design for Ubiquitous Language Learning: Mobile Phone and Interactive Television (iTV), in: *Proceedings of the IEEE International Workshop on Wireless and Mobile Technologies in Education*, IEEE Computer Society, Washington, DC, USA, pp. 85–92

[Fis54] Fisher, Ronald Aylmer: *Statistical Methods for Research Workers*, Oliver and Boyd, Edinburgh (1954)

[Fou09] Foundation for Intelligent Physical Agents: Standard Status Specifications (2009), [online, accessed 21-April-2009]

[Ful98] Fuller, Michael and Zobel, Justin: Conflation-based comparison of stemming algorithms, in: *ADCS '98: Proceedings of the Australian Document Computing Symposium*, Sydney, Australia, pp. 8–13

[Fur87] Furnas, G. W.; Landauer, T. K.; Gomez, L. M. and Dumais, S. T.: The vocabulary problem in human-system communication. *Communications of the ACM* (1987), vol. 30(11):pp. 964–971

[Gab04] Gabrilovich, Evgeniy and Markovitch, Shaul: Text categorization with many redundant features: using aggressive feature selection to make SVMs competitive with C4.5, in: *ICML '04: Proceedings of the 21st International Conference on Machine Learning*, ICML '04, ACM, New York, NY, USA, pp. 41–

[Gar08] Garfinkel, Robert; Gopal, Ram; Pathak, Bhavik and Yin, Fang: Shopbot 2.0: Integrating recommendations and promotions with comparison shopping. *Decision Support Systems* (2008), vol. 46(1):pp. 61–69

[Ge98] Ge, Niyu; Hale, John and Charniak, Eugene: A Statistical Approach to Anaphora Resolution, in: *WVLC '98: Proceedings of the 6th Workshop on Very Large Corpora*, pp. 161–170

[Gha10] Ghazanfar, Mustansar and Prugel-Bennett, Adam: An Improved Switching Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering, in: *ICDMA '10: Proceedings of the International Conference on Data Mining and Applications*

[Gol92] Goldberg, David; Nichols, David; Oki, Brian M. and Terry, Douglas: Using collaborative filtering to weave an information tapestry. *Communications of the ACM* (1992), vol. 35(12):pp. 61–70

[Got08] Gotardo, Reginaldo A.; Teixeira, Cesar A. C. and Zorzo, Sérgio D.: IP2 Model - Content Recommendation in Web-Based Educational Systems Using User's Interests and Preferences and Resources' Popularity, in: *COMPSAC '08: Proceedings of the 32nd Annual IEEE International Computer Software and Applications Conference*, IEEE Computer Society, Washington, DC, USA, pp. 460–463

[Got10] Goto, Jun; Sumiyoshi, Hideki; Miyazaki, Masaru; Tanaka, Hideki; Shibata, Masahiro and Aizawa, Akiko: Relevant TV program retrieval using broadcast summaries, in: *IUI '10: Proceeding of the 14th International Conference on Intelligent User Interfaces*, IUI '10, ACM, New York, NY, USA, pp. 411–412

[Gra02] Graham, Paul: A Plan for Spam, http://www.paulgraham.com/spam.html (2002)

[Gra03] Graham, Paul: Better Bayesian Filtering, http://www.paulgraham.com/better.html (2003)

[Gra04] Graham, Paul: *Hackers and Painters: Big Ideas from the Computer Age*, O'Reilly Media, Inc., Sebastopol, CA, USA (2004)

[Gre71] Greene, Barbara B. and Rubin, Gerald M.: Automatic Grammatical Tagging of English, Technical Report, Department of Linguistics, Brown University, Providence, Rhode Island (1971)

[Gud08] Gude, Martin; Grünvogel, Stefan M. and Pütz, Andreas: Predicting Future User Behaviour in Interactive Live TV, in: *EUROITV '08: Proceedings of the 6th European Conference on Changing Television Environments*, EUROITV '08, Springer-Verlag, Berlin, Heidelberg, pp. 117–121

[Har01] Harabagiu, Sanda M.; Bunescu, Razvan C. and Maiorano, Steven J.: Text and knowledge mining for coreference resolution, in: *NAACL '01: Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, Association for Computational Linguistics, pp. 1–8

[Hau00] Hausser, Roland: *Grundkurs Sprachwissenschaft - Mensch-Maschine-Kommunikation in natürlicher Sprache*, Springer, Berlin, Germany (2000)

[Hey08a] Heymann, Paul; Koutrika, Georgia and Garcia-Molina, Hector: Can social bookmarking improve web search?, in: *WSDM '08: Proceedings of the International Conference on Web search and Web data mining*, ACM, New York, NY, USA, pp. 195–206

[Hey08b] Heymann, Paul; Ramage, Daniel and Garcia-Molina, Hector: Social tag prediction, in: *SIGIR '08: Proceedings of the 31st annual International ACM SIGIR Conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 531–538

[Höl07] Hölbling, Günther; Rabl, Tilmann and Kosch, Harald: Intertainment, in: *MM '07: Proceedings of the 15th International Conference on Multimedia*, ACM, New York, NY, USA, pp. 475–476

[Höl08] Hölbling, Günther; Rabl, Tilmann; Coquil, David and Kosch, Harald: Interactive TV Services on Mobile Devices. *IEEE MultiMedia* (2008), vol. 15(2):pp. 72–76

[Höl10] Hölbling, Günther; Pleschgatternig, Michael and Kosch, Harald: PersonalTV - A TV recommendation system using program metadata for content filtering. *Multimedia Tools and Applications* (2010), vol. 46(2–3):pp. 259–288

[Hot06] Hotho, A.; Jäschke, R.; Schmitz, C. and Stumme, G.: Information Retrieval in Folksonomies: Search and Ranking, in: *ESWC '06: Proceedings of the 3rd European Semantic Web Conference*, vol. 4011 of *LNCS*, Springer, Heidelberg, pp. 411–426

[Hsu03] Hsu, Chih-Wei; Chang, Chih-Chung and Lin, Chih-Jen: A Practical Guide to Support Vector Classification, Technical Report, Department of Computer Science, National Taiwan University (2003), URL http://www.csie.ntu.edu.tw/~cjlin/papers.html

[Ing08] Ingason, Anton Karl; Helgadóttir, Sigrún; Loftsson, Hrafn and Rögnvaldsson, Eiríkur: A Mixed Method Lemmatization Algorithm Using a Hierarchy of Linguistic Identities (HOLI), in: *GoTAL '08: Proceedings of the 6th International Conference on Advances in Natural Language Processing*, Springer-Verlag, Berlin, Heidelberg, pp. 205–216

[Int96] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 4: MHEG registration procedure (1996), (ISO/IEC 13522-4)

[Int97a] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 1: MHEG object representation – Base notation (ASN.1) (1997), (ISO/IEC 13522-1)

[Int97b] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 3: MHEG script interchange representation (1997), (ISO/IEC 13522-3)

[Int97c] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 5: Support for base-level interactive applications (1997), (ISO/IEC 13522-5)

[Int98a] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 6: Support for enhanced interactive applications (1998), (ISO/IEC 13522-6)

[Int98b] International Organisation for Standardisation: Information technology – Generic coding of moving pictures and associated audio information – Part 6: Extensions for Digital Storage Media Command and Control (DSM-CC) (1998), (ISO/IEC 13818-6)

[Int01a] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 7: Interoperability and conformance testing for ISO/IEC 13522-5 (2001), (ISO/IEC 13522-7)

[Int01b] International Organisation for Standardisation: Information technology – Coding of multimedia and hypermedia information – Part 8: XML notation for ISO/IEC 13522-5 (2001), (ISO/IEC 13522-8)

[Int02] International Organisation for Standardisation: MPEG-7 – Multimedia Content Description Interface – (2002), (ISO/IEC 15938)

[Int05] International Organisation for Standardisation: Information technology – Multimedia framework (MPEG-21) – Part 2: Digital Item Declaration (2005), (ISO/IEC 21000-2:2005)

[Ira04] Ira, George Forman; Forman, George; Cohen, Ira and Cohen, Ira: Learning from Little: Comparison of Classifiers Given Little Training, in: *PKDD '04: Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 161–172

[Jäs07] Jäschke, Robert; Marinho, Leandro Balby; Hotho, Andreas; Schmidt-Thieme, Lars and Stumme, Gerd: Tag Recommendations in Folksonomies, in: Joost N. Kok; Jacek Koronacki; Ramon López de Mántaras; Stan Matwin; Dunja Mladenic and Andrzej Skowron (Editors) *PKDD '07: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, vol. 4702 of *LNCS*, Springer, pp. 506–514

[Joa98] Joachims, Thorsten: Text categorization with support vector machines: learning with many relevant features, in: *ECML '98: Proceedings of the 10th European Conference on Machine Learning*

[Joa99] Joachims, Thorsten: Transductive Inference for Text Classification using Support Vector Machines, in: *ICML '99: Proceedings of the 6th International Conference on Machine Learning*, ICML '99, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 200–209

[Jur00] Jurafsky, Daniel and Martin, James H.: *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Prentice Hall PTR, Upper Saddle River, NJ, USA (2000)

[Kat93] Katamba, Francis: *Morphology (Modern Linguistics)*, Palgrave, New York (1993)

[Kis06] Kiss, Tibor and Strunk, Jan: Unsupervised Multilingual Sentence Boundary Detection. *Computational Linguistics* (2006), vol. 32(4):pp. 485–525

[Kle63] Klein, Sheldon and Simmons, Robert F.: A Computational Approach to Grammatical Coding of English Words. *Journal of the ACM* (1963), vol. 10(3):pp. 334–347

[Kle03] Klein, Dan; Smarr, Joseph; Nguyen, Huy and Manning, Christopher D.: Named Entity Recognition with Character-Level Models, in: Walter Daelemans and Miles Osborne (Editors) *CoNLL '03: Proceedings of the 7th Conference on Natural Language Learning*, Edmonton, Canada, pp. 180–183

[Kos03] Kosch, Harald: *Distributed Multimedia Database Technologies supported by MPEG-7 and MPEG-21*, CRC Press, ISBN 0-8493-1854-8 (2003)

[Koy00] Koychev, Ivan and Schwab, Ingo: Adaptation to Drifting User's Interests, in: *ECML '00: Proceedings of the 11th European Conference on Machine Learning*, pp. 39–46

[Kre09] Krestel, Ralf; Fankhauser, Peter and Nejdl, Wolfgang: Latent dirichlet allocation for tag recommendation, in: *RecSys '09: Proceedings of the 3rd ACM Conference on Recommender Systems*, ACM, New York, NY, USA, pp. 61–68

[Kru98] Krupka, George R. and Hausman, Kevin: IsoQuest: Description of the NetOwl extractor system as used in MUC-7., in: *MUC-7 '98: Proceedings of the 7th Message Understanding Conference*

[Lea98] Leacock, Claudia; Chodorow, Martin and Miller, George A.: Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics* (1998), vol. 24(1):pp. 147–165

[Lei07] Leino, Juha and Räihä, Kari-Jouko: Case amazon: ratings and reviews as part of recommendations, in: *RecSys '07: Proceedings of the 2007 ACM Conference on Recommender Systems*, ACM, New York, NY, USA, pp. 137–140

[Leo02] Leopold, Edda and Kindermann, Jörg: Text Categorization with Support Vector Machines. How to Represent Texts in Input Space? *Machine Learning* (2002), vol. 46(1-3):pp. 423–444

[Les86] Lesk, Michael: Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone, in: *SIGDOC '86: Proceedings of the 5th annual International Conference on Systems documentation*, ACM, New York, NY, USA, pp. 24–26

[Li03] Li, Xiaoli and Liu, Bing: Learning to classify texts using positive and unlabeled data, in: *IJCAI'03: Proceedings of the 18th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 587–592

[Lin98] Lin, Dekang: An Information-Theoretic Definition of Similarity, in: *ICML '98: Proceedings of the 5th International Conference on Machine Learning*, ICML '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 296–304

[Liu02] Liu, Bing; Lee, Wee Sun; Yu, Philip S. and Li, Xiaoli: Partially Supervised Classification of Text Documents, in: *ICML '02: Proceedings of the 19th International Conference on Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 387–394

[Liu03] Liu, B.; Dai, Y.; Li, X.; Lee, W.S. and Yu, P.S.: Building text classifiers using positive and unlabeled examples, in: *ICDM '03: Proceedings of the 3rd IEEE International Conference on Data Mining*, pp. 179 – 186

[Lou03a] Louis, Greg: Bogofilter Calculations: Comparing Bayes Chain Rule with Fisher's Method for Combining Probabilities, http://www.bgl.nu/bogofilter/BcrFisher.html (April 2003)

[Lou03b] Louis, Greg: Testing bogofilter's calculation methods, http://www.bgl.nu/bogofilter/test6000.html (April 2003)

[Lov68] Lovins, Julie Beth: Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics* (1968), vol. 11:pp. 22 – 31

[Luo04] Luo, Xiaoqiang; Ittycheriah, Abe; Jing, Hongyan; Kambhatla, Nanda and Roukos, Salim: A mention-synchronous coreference resolution algorithm based on the Bell tree, in: *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, p. 135

[Lux08] Lux, Mathias; Granitzer, Michael and Spaniol, Marc: *Multimedia Semantics – The Role of Metadata*, vol. 101 of *Studies in Computational Intelligence*, Springer, Berlin Heidelberg (2008)

[Mac07] MacCormac, D.; Deegan, M.; Mtenzi, F. and Brendan: Implementation of a Systemto Support Mobile Computing Sessions, in: *ICPCA '07: Proceedings of the 2nd International Conference on Pervasive Computing and Applications*, pp. 521 –526

[Man02] Manuel, Roman and H., Campbell Roy: A User-Centric, Resource-Aware, Context-Sensitive, Multi-Device Application Framework for Ubiquitous Computing Environments, Technical Report, University of Illinois at Urbana-Champaign (2002)

[Man08] Manning, Christopher D.; Raghavan, Prabhakar and Schütze, Hinrich: *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA (2008)

[Mar05] Markert, Katja and Nissim, Malvina: Comparing Knowledge Sources for Nominal Anaphora Resolution. *Computational Linguistics* (2005), vol. 31(3):pp. 367–402

[Mar09] Markines, Benjamin; Cattuto, Ciro; Menczer, Filippo; Benz, Dominik; Hotho, Andreas and Stumme, Gerd: Evaluating similarity measures for emergent semantics of social tagging, in: *WWW '09: Proceedings of the 18th International Conference on World Wide Web*, ACM, New York, NY, USA, pp. 641–650

[Mar10] Martínez, José M.: MPEG-7 Overview (version 10), http://www.chiariglione.org/mpeg/ (2010)

[MB95] Meyer-Boudnik, Thomas and Effelsberg, Wolfgang: MHEG Explained. *IEEE MultiMedia* (1995), vol. 2(1):pp. 26–38

[McC95] McCarthy, Joseph F. and Lehnert, Wendy G.: Using Decision Trees for Coreference Resolution, in: *IJCAI '95: Proceedings of the 14th International Joint Conference on Artificial intelligence*, pp. 1050–1055

[McC98] McCallum, Andrew and Nigam, Kamal: A comparison of event models for Naive Bayes text classification, in: *AAAI '98: Proceedings of the Workshop on Learning for Text Categorization*, AAAI Press, pp. 41–48

[Mel01] Melville, Prem; Mooney, Raymond J. and Nagarajan, Ramadass: Content-boosted collaborative filtering, in: *SIGIR '01: Proceedings of the Workshop on Recommender Systems*

[Mih07] Mihalcea, Rada: Using Wikipedia for Automatic Word Sense Disambiguation, in: *HLT-NAACL '07: Proceedings of the main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pp. 196–203

[Mik02] Mikheev, Andrei: Periods, capitalized words, etc. *Computational Linguistics* (2002), vol. 28(3):pp. 289–318

[Mil90] Miller, George A.; Beckwith, Richard; Fellbaum, Christiane; Gross, Derek and Miller, Katherine: WordNet: An on-line lexical database. *International Journal of Lexicography* (1990), vol. 3:pp. 235–244

[Mil07] Millan, Marta; Trujillo, Maria and Ortiz, Edward: A collaborative recommender system based on asymmetric user similarity, in: *IDEAL'07: Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning*, Springer-Verlag, Berlin, Heidelberg, pp. 663–672

[Min06] Min, Okgee; Kim, Jungkun and Kim, Myungjoon: Design of an adaptive streaming system in ubiquitous environment, in: *ICACT '06: Proceedings of the 8th International Conference on Advanced Communication Technology*, vol. 2, pp. 4 pp. –1160

[Mir09] Miranda, Catarina and Jorge, Alípio Mário: Item-Based and User-Based Incremental Collaborative Filtering for Web Recommendations, in: *EPIA '09: Proceedings of the 14th Portuguese Conference on Artificial Intelligence*, Springer-Verlag, Berlin, Heidelberg, pp. 673–684

[Mis06] Mishne, Gilad: AutoTag: a collaborative approach to automated tag assignment for weblog posts, in: *WWW '06: Proceedings of the 15th International Conference on World Wide Web*, ACM, New York, NY, USA, pp. 953–954

[Mit10] Mitchell, Keith; Jones, Andrew; Ishmael, Johnathan and Race, Nicholas J.P.: Social TV: toward content navigation using social awareness, in: *EuroITV '10: Proceedings of the 8th international interactive conference on Interactive TV&Video*, EuroITV '10, ACM, New York, NY, USA, pp. 283–292

[Mor05] Morris, Steven and Smith-Chaigneau, Anthony: *Interactive TV Standards*, Focal Press, Burlington, USA (2005)

[Muk07] Mukhtar, H.; Belaid, D. and Bernard, G.: Session Mobility of Multimedia Applications in Home Networks Using UPnP, in: *INMIC '07: Proceedings of the 11th IEEE International Multi-topic Conference*, pp. 1 –6

[Nad06] Nadeau, David; Turney, Peter D. and Matwin, Stan: Unsupervised Named-Entity Recognition: Generating Gazetteers and Resolving Ambiguity., in: Luc Lamontagne and Mario Marchand (Editors) *ConfAi '06: Proceedings of the Canadian Conference on AI*, vol. 4013 of *Lecture Notes in Computer Science*, Springer, pp. 266–277

[Nad07a] Nadeau, David: Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision, http://cogprints.org/5859/1/Thesis-David-Nadeau.pdf (2007)

[Nad07b] Nadeau, David and Sekine, Satoshi: A Survey of Named Entity Recognition and Classification. *Linguisticae Investigationes* (2007), vol. 30:pp. 3–26

[Neu09] Neumann, Andreas W.: *Recommender Systems for Information Providers: Designing Customer Centric Paths to Information*, Contributions to Management Science, Springer, Heidelberg (2009)

[Ng02] Ng, Vincent and Cardie, Claire: Improving machine learning approaches to coreference resolution, in: *ACL '02: Proceedings of the 40th Annual Meeting* *on Association for Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 104–111

[Ope06] Open Mobile Alliance Ltd.: User Agent Profile – Approved Version 2.0 (2006)

[Pad05] Padgham, Von Lin and Winikoff, Michael: *Developing Intelligent Agent Systems: A Practical Guide*, John Wiley & Sons (2005)

[Pal94] Palmer, David D. and Palmer, David D.: SATZ - An Adaptive Sentence Segmentation System, Technical Report, University of California (1994)

[Pap06] Papanikolaou, Kyparisia A.; Mabbott, Andrew; Bull, Susan and Grigoriadou, Maria: Designing learner-controlled educational interactions based on learning/cognitive style and learner behaviour. *Interacting with Computers* (2006), vol. 18(3):pp. 356–384

[Pat03] Patwardhan, Siddharth; Banerjee, Satanjeev and Pedersen, Ted: Using measures of semantic relatedness for word sense disambiguation, in: *CICLing '03: Proceedings of the 4th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing'03, Springer-Verlag, Berlin, Heidelberg, pp. 241–257

[Pat07] Paterek, Arkadiusz: Improving regularized singular value decomposition for collaborative filtering, in: *SIGKDD '07: Proceedings of the KDD Cup Workshop at 13th ACM International Conference on Knowledge Discovery and Data Mining*, pp. 39–42

[Paz07] Pazzani, Michael J. and Billsus, Daniel: Content-based recommendation systems, in: Peter Brusilovsky; Alfred Kobsa and Wolfgang Nejdl (Editors) *The adaptive web*, chap. Content-based recommendation systems, Springer-Verlag, Berlin, Heidelberg (2007), pp. 325–341

[Per99] Perez, Asuncion Gomez and Benjamins, V. Richard: Overview of Knowledge Sharing and Reuse Components: Ontologies and Problem-Solving Methods, in: *BNAIC '99: Proceedings of the 11th Belgium-Netherlands Conference on Artificial Intelligence*, pp. 241 – 242

[Pla99] Platt, John C. and Platt, John C.: Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods, in: *Advances in Large Margin Classifiers*, MIT Press (1999), pp. 61–74

[Pli08] Plisson, Joël; Lavrac, Nada; Mladenic, Dunja and Erjavec, Tomaz: Ripple Down Rule learning for automated word lemmatisation. *Artificial Intelligence Communications* (2008), vol. 21(1):pp. 15–26

[Pol05] Polat, Huseyin and Du, Wenliang: SVD-based collaborative filtering with privacy, in: *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, ACM, New York, NY, USA, pp. 791–795

[Pon06] Ponzetto, Simone Paolo and Strube, Michael: Exploiting semantic role labeling, WordNet and Wikipedia for coreference resolution, in: *HLT-NAACL '06: Proceedings of the main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 192–199

[Por80] Porter, M. F.: An algorithm for suffix stripping. *Program* (1980), vol. 14(3):pp. 130–137

[Pro10] Pronk, Verus; Barbieri, Mauro; Korst, Jan and Proidl, Adolf: Integrating broadcast and web video content into personal tv channels, in: *RecSys '10: Proceedings of the 4th ACM conference on Recommender systems*, RecSys '10, ACM, New York, NY, USA, pp. 355–356

[Qiu09] Qiu, Qiang; Zhang, Yang and Zhu, Junping: Building a Text Classifier by a Keyword and Unlabeled Documents, in: *PAKDD '09: Proceedings of the 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, PAKDD '09, Springer-Verlag, Berlin, Heidelberg, pp. 564–571

[Ran08] Ransburg, Michael; Timmerer, Christian and Hellwagner, Hermann: Dynamic and Distributed Multimedia Content Adaptation based on the MPEG-21 Multimedia Framework, in: Michael Granitzer; Mathias Lux and Marc Spaniol (Editors) *Multimedia Semantics – The Role of Metadata*, vol. 101 of *Studies in Computational Intelligence*, Springer Berlin / Heidelberg (2008), pp. 3–23

[Rau05] Rauschenbach, Uwe: Interactive TV meets Mobile Computing, in: Nigel Davies; Thomas Kirste and Heidrun Schumann (Editors) *DRAG '05: Proceedings of the Seminar 05181 - Mobile Computing and Ambient Intelligence: The Challenge of Multimedia*, vol. 05181 of *Dagstuhl Seminar Proceedings*, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany IBFI, Schloss Dagstuhl, Germany

[Rei04] Reimers, Ulrich: *The Family of International Standards for Digital Video Broadcasting*, Springer, Heidelberg, Germany (2004)

[Rei05] Reimers (editor), Ulrich: *DVB : The Family of International Standards for Digital Video Broadcasting*, Springer-Verlag Berlin Heidelberg (2005)

[Rei08a] Reimers, Ulrich: *DVB – Digitale Fernsehtechnik Datenkompression und Übertragung*, Springer, Berlin, Germany (2008)

[Rei08b] Reiterer, Bernhard; Concolato, Cyril; Lachner, Janine; Feuvre, Jean Le; Moissinac, Jean-Claude; Lenzi, Stefano; Chessa, Stefano; Ferrá, Enrique Fernández; Menaya, Juan José González and Hellwagner, Hermann: User-centric universal multimedia access in home networks. *The Visual Computer* (2008), vol. 24(7-9):pp. 837–845

[Res95] Resnik, Philip: Using Information Content to Evaluate Semantic Similarity in a Taxonomy, in: *IJCAI '95: Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 448–453

[Res97] Resnick, Paul and Varian, Hal R.: Recommender systems. *Communications of the ACM* (1997), vol. 40(3):pp. 56–58

[Rey97] Reynar, Jeffrey C. and Ratnaparkhi, Adwait: A Maximum Entropy Approach to Identifying Sentence Boundaries, in: *ANLC '97: Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 16–19

[Rob02] Robinson, Gary: Spam Detection, http://radio.weblogs.com/0101454/stories/2002/09/16 /spamDetection.html (September 2002)

[Rob03] Robinson, Gary: A statistical approach to the spam problem. *Linux Journal* (2003), vol. 2003(107):p. 3

[Rob04] Robinson, Gary: Handling Redundancy in Email Token Probabilities, http://garyrob.blogs.com/handlingtokenredundancy94.pdf (May 2004)

[Roc95] Roche, Emmanuel and Schabes, Yves: Deterministic part-of-speech tagging with finite-state transducers. *Computational Linguistics* (1995), vol. 21(2):pp. 227–253

[Rog93] Roger Price: MHEG: an introduction to the future International standard for hypermedia object interchange, in: *MULTIMEDIA '93: Proceedings of the First ACM International Conference on Multimedia*, ACM Press, New York, NY, USA, pp. 121–128

[Ros00] Rosario, Barbara: Latent Semantic Indexing: An overview, Technical Report, UC Berkely School of Information (2000), URL http://www.ischool.berkeley.edu/~rosario/projects/LSI.pdf

[Ruh97] Ruhrmann, Georg and Nieland, Jörg-Uwe: *Interaktives Fernsehen*, Westdeutscher Verlag GmbH, Wiesbaden, Germany (1997)

[Rup03] Rupp, Stephan and Siegmund, Gerd: *Java in der Telekommunikation*, dpunkt.verlag, Heidelberg, Germany (2003)

[Rut96] Rutledge, Lloyd; Buford, John F. and Price, Roger: Mobile objects and the Hyoctane distributed hyperdocument server. *Computers & Graphics* (1996), vol. 20(5):pp. 633–639

[Sal86] Salton, Gerard and McGill, Michael J.: *Introduction to Modern Information Retrieval*, McGraw-Hill, Inc., New York, NY, USA (1986)

[Sal02] Salembier, Phillippe and Smith, John R.: Overview of Multimedia Description Schemes and Schema Tools, in: B. S. Manjuta; Philippe Salembier and Thomas Sikora (Editors) *Introduction to MPEG-7*, chap. 6, John Wiley & Sons, Chichester (2002), pp. 83–93

[San90] Santorini, Beatrice: Part-of-speech tagging guidelines for the Penn Treebank Project, Technical Report, Department of Computer and Information Science, University of Pennsylvania (1990)

[Sch94] Schmid, Helmut: Probabilistic Part-of-Speech Tagging Using Decision Trees, in: *Proceedings of the International Conference on New Methods in Language Processing*

[Sch95] Schmid, Helmut: Improvements In Part-of-Speech Tagging With an Application To German, in: *EACL '95 SIGDAT: Proceedings of the Special Interest Group on Linguistic data and corpus-based approaches to NLP-Workshop*, pp. 47–50

[Sch99] Schiller, Anne; Teufel, Simone; Stöckert, Christine and Thielen, Christine: Guidelines für das Tagging deutscher Textcorpora mit STTS, Technical Report, Institut fur maschinelle Sprachverarbeitung, Stuttgart (1999)

[Sch00] Schulzrinne, Henning and Wedlund, Elin: Application-layer mobility using SIP. *SIGMOBILE Mob. Comput. Commun. Rev.* (2000), vol. 4(3):pp. 47–57

[Sch01] Scholkopf, Bernhard and Smola, Alexander J.: *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, MA, USA (2001)

[Sch07] Schafer, J.; Frankowski, Dan; Herlocker, Jon and Sen, Shilad: Collaborative Filtering Recommender Systems, in: Peter Brusilovsky; Alfred Kobsa and Wolfgang Nejdl (Editors) *The Adaptive Web*, vol. 4321 of *Lecture Notes in Computer Science*, chap. 9, Springer Berlin Heidelberg, Berlin, Heidelberg (2007), pp. 291–324

[Seb02] Sebastiani, Fabrizio and Ricerche, Consiglio Nazionale Delle: Machine Learning in Automated Text Categorization. *ACM Computing Surveys* (2002), vol. 34:pp. 1–47

[Seg07] Segaran, Toby: *Programming Collective Intelligence: Building Smart Web 2.0 Applications*, O'Reilly Media, Inc., Sebastopol, CA, USA (2007)

[Sen09] Sen, Shilad; Vig, Jesse and Riedl, John: Tagommenders: connecting users to items through tags, in: *WWW '09: Proceedings of the 18th International Conference on World Wide Web*, ACM, New York, NY, USA, pp. 671–680

[She08] Shepitsen, Andriy; Gemmell, Jonathan; Mobasher, Bamshad and Burke, Robin: Personalized recommendation in social tagging systems using hierarchical clustering, in: *RecSys '08: Proceedings of the 2008 ACM Conference on Recommender systems*, ACM, New York, NY, USA, pp. 259–266

[Sig08] Sigurbjörnsson, Börkur and van Zwol, Roelof: Flickr tag recommendation based on collective knowledge, in: *WWW '08: Proceeding of the 17th International Conference on World Wide Web*, ACM, New York, NY, USA, pp. 327–336

[Soc04] Society of Motion Picture and Television Engineers: Material Exchange Format (MXF) – File Format Specification (Standard) (2004), (SMPTE 377M)

[Son02] Song, Henry; hua Chu, Hao and Kurakake, Shoji: Browser Session Preservation and Migration, in: *WWW '02: Proceedings of the 12th International Conference on World Wide Web*, pp. 7–11

[Soo01] Soon, Wee Meng; Ng, Hwee Tou and Lim, Chung Yong: A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics* (2001), vol. 27(4):pp. 521–544

[Sou05] Souvannavong, Fabrice; Mérialdo, Bernard and Huet, Benoit: Partition sampling : an active learning selection strategy for large database annotation. *IEEE Proceedings on Vision, Image and Signal Processing* (2005), vol. 152(3):pp. 347 – 355

[Spi09] Spiegel, Stephan; Kunegis, Jérôme and Li, Fang: Hydra: a hybrid recommender system [cross-linked rating and content information], in: *CNIKM '09: Proceeding of the 1st ACM International Workshop on Complex Networks meet Information & Knowledge Management*, ACM, New York, NY, USA, pp. 75–80

[Sta99] Stamatatos, E.; Fakotakis, N. and Kokkinakis, G.: Automatic Extraction of Rules for Sentence Boundary Disambiguation (1999)

[Str02] Strube, Michael; Rapp, Stefan and Müller, Christoph: The influence of minimum edit distance on reference resolution, in: *EMNLP '02: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 312–319

[Stu07] Stumpf, Simone; Rajaram, Vidya; Li, Lida; Burnett, Margaret; Dietterich, Thomas; Sullivan, Erin; Drummond, Russell and Herlocker, Jonathan: Toward harnessing user feedback for machine learning, in: *IUI '07: Proceedings of the 12th International Conference on Intelligent User Interfaces*, ACM, New York, NY, USA, pp. 82–91

[Tan08] Tang, Jie; Jin, Ruoming and Zhang, Jing: A Topic Modeling Approach and Its Integration into the Random Walk Framework for Academic Search, in: *ICDM '08: Proceedings of the 8th IEEE International Conference on Data Mining*, vol. 0, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1055–1060

[Tou07] Tougas, Jane E. and Spiteri, Raymond J.: Updating the partial singular value decomposition in latent semantic indexing. *Computational Statistics and Data Analysis* (2007), vol. 52(1):pp. 174 – 183

[Tri07] Trillo, Raquel; Ilarri, Sergio and Mena, Eduardo: Comparison and Performance Evaluation of Mobile Agent Platforms, in: *ICAS '07: Proceedings of the Third International Conference on Autonomic and Autonomous Systems*, IEEE Computer Society, Washington, DC, USA, p. 41

[TS08] Tso-Sutter, Karen H. L.; Marinho, Leandro Balby and Schmidt-Thieme, Lars: Tag-aware recommender systems by fusion of collaborative filtering algorithms, in: *SAC '08: Proceedings of the 2008 ACM symposium on Applied Computing*, ACM, New York, NY, USA, pp. 1995–1999

[UPn06] UPnP Forum: The UPnP Device Architecture 1.0 (2006)

[Vol00] Volmert, Johannes: *Grundlagen der Computerlinguistik eine Einführung in die Sprachwissenschaft für Lehramtsstudiengänge / Johannes Volmert (Hrsg.)*, Fink, München (2000)

[Wag74] Wagner, Robert A. and Fischer, Michael J.: The String-to-String Correction Problem. *Journal of the ACM* (1974), vol. 21(1):pp. 168–173

[Wei08] Weiss, Diana; Scheuerer, Johannes; Wenleder, Michael; Erk, Alexander; Gülbahar, Mark and Linnhoff-Popien, Claudia: A user profile-based personalization system for digital multimedia content, in: *DIMEA '08: Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*, ACM, New York, NY, USA, pp. 281–288

[Wit06] Witte, René and Mülle, Jutta (Editors): *Text Mining: Wissensgewinnung aus natürlichsprachigen Dokumenten*, Interner Bericht 2006-5, Universität Karlsruhe, Fakultät für Informatik, Institut für Programmstrukturen und Datenorganisation (IPD) (2006)

[Wu94] Wu, Zhibiao and Palmer, Martha: Verbs semantics and lexical selection, in: *ACL '94: Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, Association for Computational Linguistics, Morristown, NJ, USA, pp. 133–138

[Wu09] Wu, Lei; Yang, Linjun; Yu, Nenghai and Hua, Xian-Sheng: Learning to tag, in: *WWW '09: Proceedings of the 18th International Conference on World Wide Web*, ACM, New York, NY, USA, pp. 361–370

[Xu06] Xu, Jin An and Araki, Kenji: A SVM-based personal recommendation system for TV programs, in: *MMM '06: Proceedings of the 12th International Conference on Multi-Media Modelling Conference Proceedings*

[Yan03] Yang, Xiaofeng; Zhou, Guodong; Su, Jian and Tan, Chew Lim: Coreference Resolution Using Competition Learning Approach, in: *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp. 176–183

[Yan07] Yang, Xiaofeng and Su, Jian: Coreference Resolution Using Semantic Relatedness Information from Automatically Discovered Patterns, in: *ACL '07: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, The Association for Computer Linguistics

[Yu02] Yu, Hwanjo; Han, Jiawei and Chang, Kevin Chen-Chuan: PEBL: positive example based learning for Web page classification using SVM, in: *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, NY, USA, pp. 239–248

[Yu08] Yu, Jie; Luo, Xiangfeng; Xu, Zheng; Liu, Fangfang and Li, Xuhui: Representation and Evolution of User Profile in Web Activity, in: *WSCS '08: Proceedings of* *the IEEE International Workshop on Semantic Computing and Systems*, IEEE Computer Society, Washington, DC, USA, pp. 55–60

[Zas02] Zaslow, Jeffrey: If TiVo Thinks You Are Gay, Here's How to Set It Straight. *Wall Street Journal (Eastern Edition)* (2002), vol. 26 Nov:p. 1

[Zdz05] Zdziarski, Jonathan A.: *Ending Spam: Bayesian Content Filtering and the Art of Statistical Language Classification*, No Starch Press, San Francisco, CA, USA (2005)

[Zha99] Zha, Hongyuan and Simon, Horst D.: On Updating Problems in Latent Semantic Indexing. *SIAM Journal on Scientific Computing* (1999), vol. 21(2):pp. 782–791

[Zha04] Zhai, Chengxiang and Lafferty, John: A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems* (2004), vol. 22(2):pp. 179–214

[Zha09] Zhang, Ning; Zhang, Yuan and Tang, Jie: A tag recommendation system for folksonomy, in: *SWSM '09: Proceeding of the 2nd ACM workshop on Social Web Search and Mining*, ACM, New York, NY, USA, pp. 9–16

[Zho09] Zhou, Jia and Luo, Tiejian: Towards an Introduction to Collaborative Filtering, in: *CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering*, IEEE Computer Society, Washington, DC, USA, pp. 576–581

[Zhu05] Zhuang, Shelley; Lai, Kevin; Stoica, Ion; Katz, Randy and Shenker, Scott: Host mobility using an internet indirection infrastructure. *Wireless Networks* (2005), vol. 11(6):pp. 741–756

[Zim04] Zimmerman, John; Kurapati, Kaushal; Buczak, Anna L.; Schaffer, Dave; Gutta, Srinivas and Martino, Jacquelyn: TV PERSONALIZATION SYSTEM: Design of a TV Show Recommender Engine and Interface, in: Liliana Ardissono; Alfred Kobsa and Mark T. Maybury (Editors) *Personalized digital television: targeting programs to individual viewers*, Kluwer Academic Publishers (2004), pp. 27–51